From Hierarchical Partitions to Hierarchical Covers: Optimal Fault-Tolerant Spanners for Doubling Metrics

Shay Solomon [*]

Abstract

A $(1+\epsilon)$-*spanner* for a doubling metric (X,δ) is a subgraph H of the complete graph corresponding to (X,δ), which preserves all pairwise distances to within a factor of $1+\epsilon$. A natural requirement from a spanner is to be robust against node failures, so that even when some of the nodes in the network fail, the remaining part would still provide a $(1+\epsilon)$-spanner. The spanner H is called a *k-fault-tolerant* $(1+\epsilon)$*-spanner*, for any $0 \le k \le n-2$, if for any subset $F \subseteq X$ with $|F| \le k$, the graph $H \setminus F$ (obtained by removing the vertices of F, as well as their incident edges, from H) is a $(1+\epsilon)$-spanner for $X \setminus F$.

In this paper we devise an optimal construction of fault-tolerant spanners for doubling metrics. Specifically, for any n-point doubling metric, any $\epsilon > 0$, and any integer $0 \le k \le n-2$, our construction provides a k-fault-tolerant $(1+\epsilon)$-spanner with optimal degree $O(k)$ within optimal time $O(n \log n + kn)$.

We then strengthen this result to provide near-optimal (up to a factor of $\log k$) guarantees on the diameter and weight of our spanners, namely, diameter $O(\log n)$ and weight $O(k^2 + k \log n) \cdot \omega(MST)$, while preserving the optimal guarantees on the degree $O(k)$ and the running time $O(n \log n + kn)$.

Our result settles several fundamental open questions in this area, culminating a long line of research that started with the STOC'95 paper of Arya et al. and the STOC'98 paper of Levcopoulos et al.

On the way to this result we develop a new technique for constructing spanners in doubling metrics. In particular, our spanner construction is based on a novel *hierarchical cover* of the metric, whereas most previous constructions of spanners for doubling and Euclidean metrics (such as the net-tree spanner) are based on *hierarchical partitions* of the metric. We demonstrate the power of hierarchical covers in the context of geometric spanners by improving the state-of-the-art results in this area.

[*]Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot 76100, Israel. E-mail: `shay.solomon@weizmann.ac.il`. This work is supported by the Koshland Center for basic Research.

1 Introduction

1.1 Euclidean Spanners. Consider a set P of n points in $\mathbb{R}^d$ and a number $\epsilon > 0$, and let $H = (P, E)$ be a graph in which the weight $\omega(x, y)$ of each edge $(x, y) \in E$ is equal to the Euclidean distance $\|x - y\|$ between x and y. The graph H is called a $(1 + \epsilon)$*-spanner* (or simply *spanner* if the *stretch* $1 + \epsilon$ is clear from the context) for P if for every $p, q \in P$, there is a path in H between p and q whose weight (the sum of all edge weights in it) is at most $(1 + \epsilon) \cdot \|p - q\|$. Such a path is called a $(1 + \epsilon)$*-spanner path.* The problem of constructing Euclidean spanners has been studied intensively [18, 19, 47, 41, 35, 4, 5, 3, 24]; see also the treatise on Euclidean spanners and their applications by Narasimhan and Smid [39].

Euclidean spanners find applications in geometric approximation algorithms, network topology design, distributed systems, and other areas. In many applications it is required to construct a $(1 + \epsilon)$-spanner $H = (P, E)$ that satisfies some useful properties. First, the spanner should contain $O(n)$ (or nearly $O(n)$) edges. Second, the *(maximum) degree* $\mathsf{deg}(H)$ of H should be small. Third, its *weight*[1] $\omega(H) = \sum_{e \in E} \omega(e)$ should not be much greater than the weight $\omega(MST(P))$ of the minimum spanning tree $MST(P)$ of P.

A natural requirement from a spanner is to be robust against node failures, so that even when some of the nodes in the network fail, the remaining part would still provide a $(1+\epsilon)$-spanner. The spanner H is called a k*-fault-tolerant* $(1 + \epsilon)$*-spanner*, for any $0 \leq k \leq n - 2$, if for any subset $F \subseteq X$ with $|F| \leq k$, the graph $H \setminus F$ (obtained by removing the vertices of F, as well as their incident edges, from H) is a $(1 + \epsilon)$-spanner for $X \setminus F$; we will henceforth write FT as a shortcut for fault-tolerant. Note that the basic (non-FT) setting when all nodes are functioning corresponds to the case $k = 0$.

The notion of FT spanners was introduced in the pioneering work of Levcopoulos et al. [37] from STOC'98. A basic (non-FT) construction of $(1+\epsilon)$-spanners with constant degree (and thus $O(n)$ edges) and constant lightness can be built in time $O(n \log n)$ [7, 23, 31]. Levcopoulos et al. [37] observed that the $(k+1)$-power[2] of a spanner is a k-FT spanner with the same stretch. Therefore, by taking the $(k+1)$-power of the basic construction of [31], one can obtain in $O(n \log n) + n2^{O(k)}$ time, a k-FT $(1+\epsilon)$-spanner with degree and lightness both bounded by $2^{O(k)}$. Notice that any k-FT spanner (with an arbitrarily large stretch) must have degree $\Omega(k)$, and thus $\Omega(kn)$ edges; also, it is easy to see that the lightness must be $\Omega(k^2)$ [21]. Thus there is a gap between the exponential upper bound $2^{O(k)}$ and the polynomial lower bounds $\Omega(k)$ and $\Omega(k^2)$ on the degree and lightness, respectively. In addition, Levcopoulos et al. [37] devised two other constructions of k-FT spanners, one with $O(k^2 n)$ edges and running time $O(n \log n + k^2 n)$, and another with edges and running time both bounded by $O(kn \log n)$. In WADS'99 Lukovszki [38] devised two constructions of k-FT spanners, one with the optimal number of edges $O(kn)$ and with running time $O(n \log^{d-1} n + kn \log \log n)$, and another with degree $O(k^2)$.

In SoCG'03, Czumaj and Zhao [21] showed that the optimal guarantees $O(k)$ and $O(k^2)$ on the degree and lightness of k-FT spanners can be achieved using a greedy algorithm.[3] However, it is unclear whether the greedy algorithm of [21] can be implemented efficiently; a naive implementation requires time $O(n^2) \cdot Paths(n, k+1, t)$, where $Paths(n, k+1, t)$ is the time needed to check whether an n-vertex graph (with $O(kn)$ edges) contains $k + 1$ vertex-disjoint t-spanner paths between an arbitrary pair of vertices. In addition, Czumaj and Zhao [21] devised a construction of k-FT spanners with the optimal degree $O(k)$ and with lightness $O(k^2 \log n)$, within time $O(kn \log^d n + nk^2 \log k)$. (See Table 1 for a reference. We also refer to Chapter 18 in the treatise [39] for an excellent survey on Euclidean FT spanners.)

We remark that the time needed to compute a k-FT spanner is $\Omega(n \log n + kn)$. The first term $\Omega(n \log n)$ is a lower bound on the time needed to compute a basic (non-FT) spanner in the *algebraic computation tree model* [17], and the second term $\Omega(kn)$ is a lower bound on the number of edges in any k-FT spanner. Up to this date no construction of k-FT spanners could combine the optimal running time $O(n \log n + kn)$ with either the optimal number of edges $O(kn)$ (and thus the optimal degree $O(k)$)

[1]For convenience, we will henceforth refer to the normalized notion of weight $\Psi(H) = \frac{\omega(H)}{\omega(MST(P))}$, which we call *lightness.*

[2]An s-power of a graph G is a graph with the same vertex set as G, and there is an edge between any pair of vertices that are connected by a path in G with at most s edges.

[3]The $\binom{n}{2}$ edges of the underlying complete Eucldiean graph are traversed by increasing order of weight; each edge (x, y) is added to the current graph if and only if it does not contain $k + 1$ vertex-disjoint t-spanner paths between x and y.

or with the optimal lightness $O(k^2)$. In particular, the following questions are stated in the treatise of Narasimhan and Smid [39]; all these questions remained open even for 2-dimensional point sets.

Question 1 (Open Problem 26 in [39]) *Is there an algorithm that constructs a k-FT $(1+\epsilon)$-spanner with $O(kn)$ edges in $O(n \log n + kn)$ time?*

Question 2 (Open Problem 27 in [39]) *Is there an algorithm that constructs a k-FT $(1+\epsilon)$-spanner with degree $O(k)$ in $O(n \log n + kn)$ time? (This question subsumes Question 1.)*

Question 3 (Open Problem 28 in [39]) *Is there an algorithm that constructs a k-FT $(1+\epsilon)$-spanner with lightness $O(k^2)$ in $O(n \log n + kn)$ time?*

Another important measure of quality is the *(hop-)diameter* of spanners; we say that a spanner H has diameter $\Lambda(H)$ if it provides a $(1+\epsilon)$-spanner path with at most $\Lambda(H)$ edges, for every $p, q \in P$. Achieving a small diameter is desirable for some practical applications (e.g., in network routing protocols); see [6, 5, 2, 3, 11, 24], and the references therein. Euclidean Spanners that combine small diameter with some of the other parameters (among small number of edges, degree, lightness and running time) have been well studied in the last twenty years. In particular, in a seminal STOC'95 paper by Arya et al. [5], a single construction of spanners that combines all these parameters was presented, having constant degree, diameter $O(\log n)$, lightness $O(\log^2 n)$ and running time $O(n \log n)$. Moreover, Arya et al. conjectured that the lightness can be improved to $O(\log n)$, without increasing any of the other parameters; having lightness and diameter both bounded by $O(\log n)$ is optimal due to a lower bound of [24]. Arya et al.'s long-standing conjecture was resolved in the affirmative in STOC'13 by Elkin and this author [25].

Chan et al. [14] generalized the result of [25] for the FT setting. Specifically, they showed that there exist k-FT spanners with degree $O(k^2)$, diameter $O(\log n)$ and lightness $O(k^3 \log n)$. The running time of the construction of [14] was not analyzed; a naive implementation requires quadratic time. In the "Future Direction" Section of [14], Chan et al. asked whether the dependence on k in the degree and lightness bounds of their construction can be improved. A partial answer to this question was given by this author [44], who achieved degree $O(k^2)$, diameter $O(\log n)$ and lightness $O(k^2 \log n)$, within time $O(n \log n + k^2 n)$. A merging of the two manuscripts [14] and [44] gave rise to the ICALP'13 paper [15]. (See Table 1 for a reference.) We remark that the upper bounds of [14, 44, 15] are pretty far from the known lower bounds: degree $O(k^2)$ versus $\Omega(k)$, lightness $O(k^2 \log n)$ versus $\Omega(k^2)$, and running time $O(n \log n + k^2 n)$ versus $\Omega(n \log n + kn)$. In particular, the following question was left open.

Question 4 *[14, 44, 15] Is there an algorithm that constructs a k-FT spanner with degree $o(k^2)$, diameter $O(\log n)$ and lightness $O(k^2) + o(k^2 \log n)$, within time $O(n \log n) + o(k^2 n)$?*

Our Results. We show that for any set P of n points in $\mathbb{R}^d$, any $\epsilon > 0$, and any integer $0 \le k \le n-2$, one can build a k-FT $(1+\epsilon)$-spanner with optimal degree $O(k)$ within time $O(n \log n + kn)$. The running time of our construction holds in the algebraic computation tree model, and is therefore optimal as well. In particular, this result settles in the affirmative Questions 1 and 2 above.

Moreover, our spanners also achieve diameter $O(\log n)$ and lightness $O(k^2 + k \log n)$, which settles Question 4. (See Table 1 for a summary of previous and our constructions of Euclidean FT spanners.)

Consider the lightness bound. Notice that it is equal to $O(k^2)$, for all $k = \Omega(\log n)$, thus matching the naïve lower bound $\Omega(k^2)$ in this range. In other words, it provides a positive answer to Question 3 for all $k = \Omega(\log n)$. More generally, it exceeds the naïve bound $\Omega(k^2)$ by a factor of $\frac{\log n}{k}$.

However, the naïve bound $\Omega(k^2)$ on the lightness can be strengthened if we take into account the diameter $O(\log n)$ of our spanners. Indeed, any spanner with diameter $O(\log n)$ must have lightness $\Omega(\log n)$ [3, 24]. More generally, for any parameter $\rho \ge 2$, any spanner with diameter $O(\log_\rho n)$ must have lightness $\Omega(\rho \log_\rho n)$ [24]. Note also that any spanner with degree $O(\rho)$ must have diameter $\Omega(\log_\rho n)$. By combining all these lower bounds together (and substituting ρ with k), we get that one cannot do better than the following tradeoff: k-FT spanners with degree $\Omega(k)$, diameter $\Omega(\log_k n)$ and lightness $\Omega(k^2 + k \log_k n)$. Notice that the parameters of our spanners are pretty close to this lower bound tradeoff, with only a slack of $\log k$ on the diameter and a slack of $\min\{\log k, \frac{\log n}{k}\} = O(\log \log n)$ on the lightness.

Reference	# Edges	Degree	Diameter	Lightness	Running Time	Metric
[37]	$n2^{O(k)}$	$2^{O(k)}$	unspecified	$2^{O(k)}$	$O(n \log n) + n2^{O(k)}$	Euclidean
[37]	$O(k^2 n)$	unspecified	unspecified	unspecified	$O(n \log n + k^2 n)$	Euclidean
[37]	$O(kn \log n)$	unspecified	unspecified	unspecified	$O(kn \log n)$	Euclidean
[38]	$O(kn)$	unspecified	unspecified	unspecified	$O(n \log^{d-1} n + kn \log \log n)$	Euclidean
[21]	$O(kn)$	$O(k)$	unspecified	$O(k^2)$	$O(n^2) \cdot Paths(n, k+1, t)$	Euclidean
[21]	$O(kn)$	$O(k)$	unspecified	$O(k^2 \log n)$	$O(kn \log^d n + nk^2 \log k)$	Euclidean
[13]	$O(kn)$	unspecified	unspecified	unspecified	unspecified	doubling
[13]	$O(km)$	unspecified	$O(\alpha(m, n))$	unspecified	unspecified	doubling
[13]	$O(k^2 n)$	$O(k^2)$	unspecified	unspecified	unspecified	doubling
[14, 44, 15]	$O(k^2 n)$	$O(k^2)$	$O(\log n)$	$O(k^2 \log n)$	$O(n \log n + k^2 n)$	doubling
New	$\boldsymbol{O(kn)}$	$\boldsymbol{O(k)}$	$\boldsymbol{O(\log n)}$	$\boldsymbol{O(k^2 + k \log n)}$	$\boldsymbol{O(n \log n + kn)}$	**doubling**

Table 1: A comparison between the previous state-of-the-art and our constructions of FT spanners for low-dimensional Euclidean and doubling metrics. See Theorem 1.1 for the dependencies of our construction on ϵ and d.

1.2 Doubling Metrics. The *doubling dimension* of a metric (X, δ) is the smallest value d such that every ball B in the metric can be covered by at most 2^d balls of half the radius of B. This notion generalizes the Euclidean dimension, since the doubling dimension of the Euclidean space $\mathbb{R}^d$ (equipped with any of the ℓ_p norms) is $\Theta(d)$. A metric is called *doubling* if its doubling dimension is constant. Doubling metrics were implicit in the works of Assoud [8] and Clarkson [20], and were explicitly defined by Gupta et al. [32]. Since then they have been studied intensively (see, e.g., [36, 46, 33, 11, 42, 1, 9, 15]).

Spanners for doubling metrics were also subject of intensive research [26, 12, 11, 33, 40, 29, 30, 42, 13, 25, 14, 44, 15]. In many of these works the objective is to devise spanners that achieve one parameter or more among small number of edges, degree, diameter, lightness, and running time. Spanners for doubling metrics were also found useful for approximation algorithms [9] and for machine learning [28].

The literature on FT spanners for doubling metrics is quite sparse, and consists of the following papers [13, 14, 44, 15]. In ICALP'12, Chan et al. [13] devised three constructions of k-FT spanners for doubling metrics. Their first construction achieves the optimal number of edges $O(kn)$, the second achieves $O(km)$ edges and diameter $O(\alpha(m, n))$, where α is the two-parameter inverse-Ackermann function, and the third construction achieves degree $O(k^2)$. The other papers are the follow-ups [14, 44, 15] to [25] that were discussed in Section 1.1, which provide within $O(n \log n + k^2 n)$ time, a k-FT spanner with degree $O(k^2)$, diameter $O(\log n)$ and lightness $O(k^2 \log n)$. The constructions of [14, 44, 15] apply to doubling metrics.

Obviously, Questions 1-4 from Section 1.1 are relevant for doubling metrics as well. Moreover, even much weaker variants of these questions can be asked. In particular, the previous state-of-the-art bounds on the degree and lightness of k-FT spanners for doubling metrics are $O(k^2)$ and $O(k^2 \log n)$, respectively; hence it is natural to ask if these bounds can be improved, regardless of the running time issue.

We demonstrate that our construction extends to doubling metrics without incurring any overhead (beyond constants) in the degree, diameter, lightness, and running time. Notice that our construction improves all the previous state-of-the-art constructions of FT spanners for both Euclidean and doubling metrics, disregarding the construction of [21] which requires a somewhat unreasonable running time of $O(n^2) \cdot Paths(n, k+1, t)$. (See Table 1 for a summary of previous and our results.)

In particular, our result settles Questions 1, 2 and 4 in the affirmative for doubling metrics.

Question 3 is settled for all $k = \Omega(\log n)$, but remains open in the complementary range of $k = o(\log n)$). We remark that the problem of constructing spanners for doubling metrics with sub-logarithmic lightness is a central open question in this area (see [9]), even for basic (non-FT) spanners, and regardless of the diameter or any other parameter of the construction. In other words, a positive solution to Question 3 in the range $o(\log n)$ seems currently out of reach for doubling metrics, even for the basic case $k = 0$.

The main result of this paper is summarized in the following theorem.

Theorem 1.1 *Let (X, δ) be an n-point doubling metric, with an arbitrary doubling dimension d. For any $\epsilon > 0$ and any integer $0 \leq k \leq n-2$, a k-FT $(1+\epsilon)$-spanner with degree $\epsilon^{-O(d)} \cdot k$, diameter $O(\log n)$ and lightness $\epsilon^{-O(d)}(k^2 + k \log n)$ can be built within $\epsilon^{-O(d)}(n \log n + kn)$ time.*

1.3 Our and Previous Techniques. Most previous works on geometric FT spanners [37, 38, 21, 39] apply to Euclidean metrics, and rely heavily on profound geometric properties of low-dimensional Euclidean metrics, such as the *gap property* [16, 7] and the *leapfrog property* [22]. In particular, these constructions do not extend to arbitrary doubling metrics. In contrast, in our work we only use standard packing arguments, and do not rely on any other geometric property of low-dimensional Euclidean metrics. Consequently, our construction applies to arbitrary doubling metrics.

On the other hand, there are some similarities between our techniques and the techniques used in the previous works on FT spanners for doubling metrics [13, 14, 44, 15]. The starting point in both our work and the works of [13, 14, 44, 15] is the standard *net-tree spanner* of [26, 12], which is induced from a *net-tree* $T = T(x)$ that corresponds to a *hierarchical partition* of the metric (X, δ). Any tree node x is associated with a single point $p(x)$ that belongs to the point set $D(x)$ of its descendant leaves. For any pair of tree nodes at the same level that are close together (with respect to the distance scale at that level), a *cross edge* is added. The net-tree spanner is the union of the tree edges (i.e., the edges of T) and the cross edges. (See Section 2 for more details.) To obtain a k-FT spanner, [13, 14] use $k+1$ net-trees rather than one, and in each of these trees, each tree node x is associated with a single point from $D(x)$. Another idea that is used in [44, 15] is to use a single net-tree, but to associate each tree node x with $\Theta(k)$ points from $D(x)$ (if any) rather than a single point. To achieve degree $O(k^2)$, [13, 14] used single-sink spanners similar to those in [5], whereas [44, 15] used a rerouting technique from the bounded degree spanner construction of [30]. However, achieving degree $o(k^2)$ using these ideas is doomed to failure. First, in [13, 14] there are $k+1$ net-trees, each of which contributes $\Omega(k)$ units to the degree load, thereby giving a total degree of $\Omega(k^2)$. Incurring a degree of $\Omega(k^2)$ from the approach of [44, 15] is also inevitable. The main problem is that a node x is associated with $\Theta(k)$ points from $D(x)$ (if any). Consequently, the same leaf point p may belong to $\Theta(k)$ different sets $D(x)$ of internal nodes x. For each cross edge that is incident on any of these $\Omega(k)$ nodes, we must connect p with $\Omega(k)$ edges, and so the degree of p becomes $\Omega(k^2)$.

The basic idea of associating nodes of net-trees and other hierarchical tree structures (such as split trees and dumbbell trees) with points from their descendant leaves has been widely used in the geometric spanner literature; see [10, 5, 26, 12, 11, 39, 30, 13], and the references therein. In particular, any point in $D(x)$ is close to the original net-point $p(x)$ with respect to the distance scale of x, denoted $rad(x)$.

Instead of restricting ourselves to $D(x)$, we suggest to look at a larger set $B(x)$ of all points that belong to the ball of radius $O(rad(x))$ centered at $p(x)$. By associating nodes x with points from $B(x)$, we get a *hierarchical cover* of the metric. Since the doubling dimension is constant, this cover will have a constant *degree* (every point will belong to a constant number of sets $B(\cdot)$ at each level). We use this constant degree hierarchical cover to obtain degree $O(k)$ as follows. A node x will appoint to itself $\Theta(k)$ points from $B(x)$ whose degree due to cross edges at lower levels is small enough; let $S(x)$ be the set of these points, called *surrogates*. A cross edge (x, y) increases the degree of surrogates of $S(x)$. However, every unit of increase to a surrogate $p \in S(x)$ is due to some surrogate $q \in S(y)$ that is close to p–the distance between p and q will be $O(\frac{1}{\epsilon} \cdot rad(x))$. This surrogate q, as well as other points that are within distance $O(rad(x))$ from q, will not necessarily be in $B(x)$. However, since the distance scales increase exponentially with the level of a tree node, all these points must belong to a *close ancestor* x' of x, where the level of x' is higher than that of x by at most $O(\log \frac{1}{\epsilon})$ units. The key idea here is to "compensate" one of the close ancestors x' of x at a value that matches the "cost" of x due to cross edges. By re-appointing the same surrogates of $S(x)$ for $O(\log \frac{1}{\epsilon})$ levels continuously (at which stage we must reach a close ancestor x' of x that is compensated), and only then appointing new surrogates for $S(x')$, we will be able to achieve the optimal degree $O(k)$. While in the approach of [44, 15] each point of X belongs to lists $D(x)$ of $\Omega(k)$ nodes x, our approach can guarantee (though this does not follow immediately from the above discussion) that each point of X will serve as a surrogate of only $O(\log \frac{1}{\epsilon})$ nodes. Having assigned surrogates for each node, we replace each edge (x, y) of the basic spanner by a bipartite clique between the corresponding surrogate sets $S(x)$ and $S(y)$. Observe that $S(x)$ and $S(y)$ may contain $\Theta(k)$ points each, and so the size of this bipartite clique may be as large as $\Theta(k^2)$. In other words, we may replace a single edge of the basic spanner by $\Theta(k^2)$ edges. Since the basic spanner has $\Theta(n)$ edges, one might get the *wrong* impression that our FT spanner contains $\Theta(k^2 n)$ edges, which is far too much (by a factor of k)! There are two reasons why the number of edges in our FT spanner is in check. First, it turns out that many edges (x, y) of the basic spanner satisfy $|S(x)| + |S(y)| \ll k$, which implies that there are many sparse bipartite cliques. More specifically, we show that the overall number of edges in all these sparse bipartite cliques is $O(kn)$. Second, even though there are still many edges in the basic spanner which satisfy $|S(x)| + |S(y)| = \Theta(k)$, we demonstrate that most of them are in fact *redundant*, and such redundant edges need not be replaced by bipartite cliques. (See Section 3.1 for more details.)

The main challenge of this work was to achieve the optimal bound $O(k)$ on the degree. However, small degree (which also implies small number of edges) is just one out of four parameters that are of interest to us. Our construction is tailored in such a way that once degree $O(k)$ is achieved, the rest of the parameters come almost for free.

Lightness. The standard net-tree analysis shows that the basic spanner construction has lightness $O(\log n)$. Replacing each edge of the basic spanner with a bipartite clique of size $O(k^2)$ between the corresponding surrogate sets gives rise to lightness $O(k^2 \log n)$. Improving the lightness bound to $O(k^2+k\log n)$ is more involved. The idea is to try replacing each edge (x, y) of the basic spanner with a bipartite matching instead of a bipartite clique; this is possible only when both $S(x)$ and $S(y)$ contain $\Theta(k)$ surrogates, which is not always the case (see Section 4). The total lightness of the bipartite matchings is clearly $O(k \log n)$. The more interesting part is to show that the total lightness of the bipartite cliques is $O(k^2)$.

Diameter. We can achieve diameter $O(\log n)$ by shortcutting the net-tree using the 1-spanners of [45]. To control the lightness, we follow an idea from [14, 44, 15] and shortcut the *light subtrees* (those at small enough distance scales). If this is done carelessly, the degree and lightness will increase to at least $O(k^2)$ and $O(k^2 \log n)$, respectively. However, by pruning a certain subset of nodes from the net-tree, and using bipartite matchings for the shortcut edges whenever possible, the desired bounds can be achieved.

Running time. Given the standard net-tree spanner construction and the 1-spanners of [45], our construction can be implemented within $O(kn)$ time in a rather straightforward manner. Since the standard net-tree spanner can be built within time $O(n \log n)$ [40, 30], and since the same amount of time suffices to build the 1-spanners of [45], the overall running time of our construction is $O(n\log n+kn)$. We remark that for low-dimensional Euclidean metrics, some variants of the net-tree (such as the fair split tree of [10]) can be built within time $O(n \log n)$ in the algebraic computation tree model. Consequently, the basic spanner construction can also be built within time $O(n \log n)$ in this model. The time bound $O(n \log n)$ for the 1-spanners of [45] also applies to this model. Therefore, for low-dimensional Euclidean metrics, the time bound of our construction applies to this model as well. For arbitrary doubling metrics, one should also need a rounding operation (to allow one to find the i for which $2^i < x \leq 2^{i+1}$, for any x) [27].

1.4 Similar Results by Kapoor and Li. Independently of our work, Kapoor and Li [34] obtained similar results for Euclidean FT spanners. In particular, they showed that for any low-dimensional Euclidean metric, one can build within time $O(n \log n + kn)$, a k-FT $(1 + \epsilon)$-spanner with degree $O(k)$ and lightness $O(k^2)$. Similarly to our result, their running time and degree bounds are optimal. In addition, they obtain the optimal lightness bound $O(k^2)$, while our lightness bound $O(k^2 + k \log n)$ has a slack of $\frac{\log n}{k}$. However, while the diameter of their spanners is unbounded, our spanners have diameter $O(\log n)$. Moreover, their result relies heavily on geometric properties of low-dimensional Euclidean metrics and thus cannot be extended to doubling metrics, while our result applies to all doubling metrics. In particular, the techniques and ideas used in the two papers are inherently different.

It should be noted that the results of Kapoor and Li [34] were obtained much prior to our results.[4] However, we stress that our results were achieved before this author became aware of the results of [34].

1.5 Organization. In Section 2 we define the basic notions and present the notation that is used throughout the paper. Section 3 is devoted to our construction of FT spanners with optimal degree. More specifically, the description of the construction is given in Section 3.1, whereas its analysis is given

[4]Private communication with Sanjiv Kapoor, March 2013.

in Section 3.2 (degree analysis) and Section 3.3 (fault-tolerance and stretch analysis). In Sections 4 and 5 we show that small lightness and diameter, respectively, can be obtained using a few simple modifications to the construction of Section 3. The running time issue is addressed in Section 6.

2 Preliminaries

Let (X, δ) be an arbitrary n-point doubling metric. Without loss of generality, we assume that the minimum inter-point distance of X is equal to 1. We denote by $\Delta = \max_{u,v \in X} \delta(u, v)$ the *diameter* of X.

A set $Y \subseteq X$ is called an r-*cover* of X if for any point $x \in X$ there is a point $y \in Y$, with $\delta(x, y) \leq r$. A set Y is an r-*packing* if for any pair of distinct points $y, y' \in Y$, it holds that $\delta(y, y') > r$. For $r_1 \geq r_2 > 0$, we say that a set $Y \subseteq X$ is an (r_1, r_2)-*net* for X if Y is both an r_1-cover of X and an r_2-packing; such a net can be constructed by a greedy algorithm. We will use the following standard packing argument.

Fact 2.1 *Let $R \geq 2r > 0$ and let $Y \subseteq X$ be an r-packing in a ball of radius R. Then, $|Y| \leq (\frac{R}{r})^{2\dim}$.*

Hierarchical Nets. We consider the hierarchical nets that are used by Gottlieb and Roditty [30]. Write $\ell = \lceil \log_5 \Delta \rceil$, and let $\{N_i\}_{i \geq 0}^{\ell}$ be a sequence of hierarchical nets, where $N_0 = X$ and for each $i \in [\ell]$, N_i is a $(3 \cdot 5^i, 5^i)$-net for N_{i-1}.[5] For each $i \in [\ell]$, we refer to N_i as the *i-level net* and the points of N_i are called the *i-level net points*. Note that $N_0 = X \supseteq N_1 \supseteq \ldots \supseteq N_\ell$, and N_ℓ contains exactly one point. The same point of X may have instances in many nets; specifically, an i-level net point is necessarily a j-level net point, for every $j \in [0, i]$. When we wish to refer to a specific instance of a point $p \in X$, which is determined uniquely by some level $i \in [0, \ell]$ (such that $p \in N_i$), we may denote it by a pair (p, i).

Net-tree. The hierarchical nets induce a hierarchical tree $T = T(X)$, called *net-tree* [26, 12].
Each node in the net-tree T corresponds to a unique net-point; we will use (p, i) to denote both the net point and the corresponding tree node. We refer to the nodes corresponding to the i-level net points as the *i-level nodes*. The only ℓ-level node is the root of T, and for each $i \in [\ell]$, each $(i-1)$-level node has a parent at level i within distance $3 \cdot 5^i$; such a node exists since N_i is a $3 \cdot 5^i$-cover for N_{i-1}. (We may assume that for a pair $(p, i+1), (p, i)$ of net-points, $(p, i+1)$ will be the parent of (p, i) in T.) Thus, any descendant of every i-level node can reach it by climbing a path of weight at most $\sum_{j \in [i]} 3 \cdot 5^i \leq 4 \cdot 5^i$. By Fact 2.1, since the doubling dimension is constant, each node has only a constant number of children. For a tree node $v = (p, i)$, let $Ch(v)$ denote the set of children of v in the net-tree T.

Net-tree Spanner via Cross Edges. We recap the basic spanner construction $H = H(X)$ of [26, 12]. In order to achieve stretch $(1 + \epsilon)$, for each $i \in [0, \ell - 1]$, *cross edges* are added between i-level nodes that are close together with respect to the distance scale 5^i at that level. Specifically, for any pair $(p, i), (q, i)$ of distinct i-level nodes such that $\delta(p, q) \leq \gamma 5^i$, for some parameter $\gamma = \Theta(\frac{1}{\epsilon})$, we add a cross edge between (p, i) and (q, i) of weight $\delta(p, q)$. The basic spanner construction H is obtained as the union of the *tree edges* (those connecting nodes with their children in T) and the cross edges, where an edge between a pair of nodes is translated to an edge between the corresponding points.

By Fact 2.1, the degree of any tree node in the spanner H (due to tree and cross edges) is $\epsilon^{-O(d)}$. Moreover, it can be shown that H has $n \cdot \epsilon^{-O(d)}$ edges [26, 12]. On the other hand, as the same point may have instances in many nets (and thus in many tree nodes), the degree of a point may be unbounded.

The following lemma from [26, 12] gives the essence of the basic cross edges framework.

Lemma 2.2 (Basic Cross Edges Framework Guarantees Low Stretch) *Consider the basic spanner construction H, whose edge set consists of all the tree edges and the cross edges (defined with respect to some parameter $\gamma = O(\frac{1}{\epsilon})$). For each $p, q \in X$, the spanner H contains a $(1 + \epsilon)$-spanner path $\Pi_{p,q}$, obtained by climbing up from the leaf nodes $(p, 0)$ and $(q, 0)$ to some j-level ancestors (p', j) and (q', j), respectively, where (p', j) and (q', j) are connected by a cross edge and $\delta(p, q) = \Theta(\frac{1}{\epsilon}) \cdot 5^j$.*

[5] As mentioned in [30], this choice of parameters is needed in order to achieve running time $O(n \log n)$.

As mentioned above, the degree of the spanner H is unbounded. To reduce the degree of the spanner, we will appoint to each internal node of the net-tree T a *surrogate*, which is a point in X that is nearby (with respect to the distance scale at that level). More specifically, a node (p, i) of T, $i \in [\ell]$, will be appointed a surrogate $q \in X$ such that $\delta(p, q) = O(5^i)$; the surrogate of a node (p, i) will be denoted by $s(p, i)$. (For technical convenience, we define the surrogate of a leaf node $(p, 0)$ as p itself, i.e., $s(p, 0) = p$.) Another important requirement from a surrogate of an i-level node is that its degree due to cross edges at lower levels $j \in [0, i-1]$ would be sufficiently small; we will discuss this issue in detail later on.

Given a surrogate for each tree node, the basic spanner construction H is translated to its *surrogate spanner* $s(H)$ by replacing each tree edge $e = ((p, i+1), (q, i))$ (respectively, cross edge $e = ((p, i), (q, i)))$ with its *surrogate edge* $s(e) = (s(p, i+1), s(q, i))$ (resp., $s(e) = (s(p, i), s(q, i))$). We will demonstrate (Section 3) that surrogates can be appointed to nodes, so that the resulting surrogate spanner has constant degree. Moreover, by increasing the number of surrogates with which each node is appointed from 1 to (at most) $k+1$, and thus replacing each edge of the basic spanner H by a bipartite clique between the corresponding surrogates, we will obtain a k-FT spanner with optimal degree $O(k)$.

Lemma 2.3 (Extended Cross Edges Framework Guarantees Low Stretch) *For each $p, q \in X$, consider the $(1+\epsilon)$-spanner path $\Pi_{p,q}$ in H that is guaranteed by Lemma 2.2. The corresponding* surrogate path *$s(\Pi_{p,q})$ in $s(H)$, obtained by replacing each edge e of $\Pi_{p,q}$ by its surrogate edge $s(e)$, is a $(1+O(\epsilon))$-spanner path (between the same endpoints p and q). We can reduce the stretch of the path from $1+O(\epsilon)$ back to $1+\epsilon$ by scaling γ up by an appropriate constant.*

3 Fault-Tolerant Spanners with Optimal Degree

In this section we devise a construction of k-FT spanners with optimal degree $O(k)$. The description of the construction is given in Section 3.1, whereas its analysis is given in Section 3.2 (degree analysis) and Section 3.3 (fault-tolerance and stretch analysis).

3.1 The Construction

3.1.1 The Basic Sceheme. Recall that the weight of i-level cross edges in the basic spanner construction H is at most $\gamma 5^i$, where $\gamma = O(\frac{1}{\epsilon})$, and define $\tau = \lceil \log_5 \gamma \rceil + 1$. Also, let $\xi = \xi(\epsilon, dim) = \epsilon^{-O(d)}$ be an upper bound for the maximum degree of any tree node in the basic spanner H.

The general approach is to try appointing $k+1$ surrogates $s_1(p, i), \ldots, s_{k+1}(p, i)$ for each tree node (p, i), so that for any k node failures in the network, at least one surrogate of (p, i) will be functioning. The set $S(p, i) = \{s_1(p, i), \ldots, s_{k+1}(p, i)\}$ will be called the *surrogate set* of (p, i). A surrogate $s \in S(p, i)$ of an i-level node (p, i) is a point q at distance $\delta(p, q) = O(5^i)$ from p, whose degree due to cross edges at lower levels $j \in [0, i-1]$ is at most $D = (\tau + 4)\xi^2(2k+1)$; we henceforth refer to D as the *degree threshold*.

The basic spanner H is given as the union of the tree and cross edges. The FT-spanner construction $\mathcal{H}$ is obtained from H by replacing each edge of H with a bipartite clique. Specifically, every cross edge $((p, i), (q, i))$ is replaced with a bipartite clique between the corresponding surrogate sets $S(p, i)$ and $S(q, i)$. Such a cross edge $((p, i), (q, i))$ is referred to as an *i-level cross edge* (of the basic spanner H). Moreover, we may also refer to the edges of the corresponding bipartite clique as cross edges (of the FT spanner $\mathcal{H}$). Similarly, a tree edge $((p, i+1), (q, i))$ is replaced with a bipartite clique between $S(p, i+1)$ and $S(q, i)$; such a tree edge $((p, i+1), (q, i))$ is referred to as an *i-level tree edge* (of the basic spanner H), and we may also refer to the edges of the corresponding bipartite clique as tree edges (of the FT spanner $\mathcal{H}$). Roughly speaking, each bipartite clique that replaces an i-level tree edge is incarnated in a clique that replace some i-level or $(i+1)$-level cross edge; refer to Section 3.2.3 for a rigorous argument. For the intuitive discussion of this section we will henceforth restrict our attention to cross edges.

Next, we describe how to compute the surrogate sets, which is the crux of the problem.

For a node (p,i), denote by $D(p,i)$ its *descendant set*, i.e., the set of points in its descendant leaves. Note that for each point $q \in D(p,i)$, we have $\delta(p,q) \leq 4 \cdot 5^i$. In particular, any point $q \in D(p,i)$ is at a small enough distance from p to serve as (p,i)'s surrogate. If there are less than $k+1$ points in $D(p,i)$ whose degree is at most D, we need to look for surrogates outside $D(p,i)$; in particular, we can look for nearby i-level nodes (q,i), with $\delta(p,q) = O(5^i)$, and try to use somehow their descendant sets $D(q,i)$. We remark that in order to achieve k-fault-tolerance, it is sufficient that the surrogate set $S(p,i)$ of a node (p,i) will contain its entire descendant set $D(p,i)$, even if $|D(p,i)| \ll k+1$. Nevertheless, we will see later that even in such cases it is still advantageous for a node to have $k+1$ surrogates, if possible.

The surrogate sets are computed bottom-up, starting at the leaf nodes. For a leaf $(p,0)$, it is enough to take just p to the surrogate set $S(p,0)$, i.e., $S(p,0) = D(p,0) = \{p\}$. More generally, we can take the surrogate set $S(p,i)$ to be $D(p,i)$, for small levels $i = O(1)$ in the tree. However, as we climb up the tree, we might not be able to guarantee that $S(p,i)$ contains the entire descendant set $D(p,i)$ due to degree violations; in this case it is critical to guarantee that $|S(p,i)| = k+1$. For any point $a \in D(p,i)$ that cannot be taken to the surrogate set $S(p,i)$, its degree must exceed the degree threshold D, which, in turn, implies that many cross edges at lower levels are incident on nodes of which a is a surrogate; more specifically, since each node may be incident on at most ξ cross edges at each level (in the basic spanner H) and D is sufficiently large, it follows that some cross edges at levels smaller than $i - \tau$ must be incident on nodes of which a is a surrogate. Intuitively, any j-level cross edge between a j-level node $(\tilde{a}, j)$ for which $a \in S(\tilde{a}, j)$ and some other j-level node (q,j), where $j \leq i - \tau$, should give rise to additional points that are close to a and thus also to p, namely, those in $D(q,j)$. Indeed, first note that the distance between $\tilde{a}$ and its surrogate a is $O(5^j)$. Second, the weight of j-level cross edges is at most $\gamma 5^j$, and so $\delta(a,q) \leq O(5^j) + \gamma 5^j \leq O(5^j) + \gamma 5^{i-\tau} = O(5^i)$. Hence for any point $b \in D(q,j)$,

$$\delta(p,b) \;\leq\; \delta(p,a) + \delta(a,q) + \delta(q,b) \;\leq\; 4 \cdot 5^i + O(5^i) + 4 \cdot 5^j \;=\; O(5^i).$$

This means that we can take the points in $D(q,j)$ of small degree to the surrogate set $S(p,i)$ of (p,i); however, if there are points in $D(q,j)$ whose degree exceed D, we will try to apply this argument again. Consequently, a surrogate of a node (p,i) need not belong to $D(p,i)$, but rather to a (possibly much larger) set $F(p,i)$ which we refer to as the *friend set* of p. This set $F(p,i)$ will contain $O(k)$ points of small degree that are at distance $O(5^i)$ from p (hence $F(p,i)$ is some subset of $O(k)$ points from the *ball set* $B(p,i)$ mentioned in Section 1.3). We need to show that whenever the degrees of surrogates of some node (p,i) are about to exceed D, there are at least $k+1$ points in $F(p,i)$ of small degree which can be appointed as new surrogates instead of the old ones, and so the degree bound will always be in check.

For now assume that the degree of a node (p,i) increases only due to cross edges (the degree contribution due to tree edges can be easily controlled, as we shall later see). Consider a cross edge $((p,i),(q,i))$, which gives rise to a bipartite clique between $S(p,i)$ and $S(q,i)$, and thus increases the degree of each point in $S(p,i)$ by $|S(q,i)|$ units. We will show that $|F(q,i)| \geq |S(q,i)|$, and the key idea is to let (q,i) share its friend set with (p,i). In this way we compensate (p,i) at a value that matches its cost due to cross edges. However, note that p and q may be at distance $\gamma 5^i$ apart (recall that $\gamma 5^i$ is the upper bound on the weight of i-level cross edges), and so the points in $F(q,i)$ may be too far from (p,i) to serve as its friends. Thus the points of $F(q,i)$ are not moved immediately to $F(p,i)$, but rather to a temporary set $R(p,i)$ called the *reserve set* of (p,i). All the points in $R(p,i)$ will be moved to $F(p,i)$ within τ levels. During these τ levels in which some point in the reserve set is too far to become a friend, we are going to re-appoint the same surrogates over and over. That is, new surrogates of a node are appointed for a long *term* of $\Omega(\tau)$ levels, which means that no ancestor of such a node in the next $\Omega(\tau)$ levels will appoint new surrogates. During this term of $\Omega(\tau)$ levels, the points of $R(p,i)$ become closer and closer to the respective ancestor of (p,i), until they are close enough to become its friends and thus appointed as its surrogates–at this stage new surrogates are appointed to that ancestor, also for a term of $\Omega(\tau)$ levels.

As mentioned in Section 1.3, one may get the *wrong* impression that our FT spanner $\mathcal{H}$ contains $O(k^2 n)$ edges rather than $O(kn)$ edges. Indeed, each edge (x,y) of the basic spanner H is translated into a bipartite clique in $\mathcal{H}$ between the corresponding surrogate sets $S(x)$ and $S(y)$. Since $S(x)$ and $S(y)$

may contain $\Theta(k)$ points each, the size of such a bipartite clique may be as large as $\Theta(k^2)$. Recalling that the basic spanner has $\Theta(n)$ edges, it may seem that our FT spanner $\mathcal{H}$ contains $\Theta(k^2 n)$ edges, which is far too much! There are two reasons why the number of edges in our FT spanner is in check. First, it turns out that many nodes x have a small surrogate set $S(x)$; such a node x is called *small* (see Section 3.1.2 for more details). Consequently, many edges (x, y) of the basic spanner satisfy $|S(x)| + |S(y)| \ll k$, which implies that many of the bipartite cliques are sparse. Second, even though there are still many edges (x, y) in the basic spanner which satisfy $|S(x)| + |S(y)| = \Theta(k)$, we demonstrate that most of them are in fact *redundant*, and such redundant edges need not be replaced by bipartite cliques. The issue of redundant edges is related to the notions of leeches and hosts, and is discussed in detail in Section 3.1.3.

3.1.2 Complete Versus Incomplete, Small Versus Large, Clean Versus Dirty. A node (p, i) is called *complete* if $|S(p, i)| \geq k + 1$; otherwise $|S(p, i)| < k + 1$, and it is *incomplete*. A node is called *large* if $|F(p, i) \cup S(p, i)| \geq 2k + 2$; otherwise $|F(p, i) \cup S(p, i)| < 2k + 2$, and it is *small*. Since any point of $F(p, i)$ can serve as a surrogate, our construction guarantees that any large node must be complete. The friend set $F(p, i)$ of (p, i) will contain $O(k)$ (more specifically, at most $3k + 3$) clean *10-friends* of (p, i); we say that a point q or a node (q, i) is a t-friend of some i-level node (p, i) if $\delta(p, q) \leq t \cdot 5^i$.

Small and large nodes are handled differently. For a small node (p, i), all points in $D(p, i)$ are appointed as surrogates, i.e., we assign $S(p, i) = D(p, i)$. We will show later that $D(p, i) \subseteq F(p, i)$.[6] This means that a small node must have less than $2k + 2$ surrogates. Even though we may upper bound the number of surrogates by $k+1$, we do not attempt to do this; it turns out that allowing more than $k+1$ (but up to $2k+2$) surrogates is useful for simplifying the analysis to some extent. Assuming by induction that $S(x) = D(x)$ holds for every child x of (p, i), we have $S(p, i) = D(p, i) = \bigcup_{x \in Ch(p,i)} D(x) = \bigcup_{x \in Ch(p,i)} S(x)$. In other words, the old surrogates (from the surrogate sets of (p, i)'s children) are re-used.

Note that a small node re-uses the surrogates of its children. Similarly to small nodes, we would like a large node to re-use the surrogates of (one of) its children. However, sometimes a large node must appoint new surrogates (this happens quite rarely). Specifically, an appointment of new surrogates to a large node x is made for a long *term* of at least $\tau + 3 = \Omega(\tau)$ levels; recall that $\tau = \lceil \log_5 \gamma \rceil + 1$. During this term, each ancestor of x' of x would re-use the same surrogates of x (in fact, x' may choose to use the surrogates of another descendant of x'); we call x an *appointing node*. The surrogates $S(x)$ of x are chosen arbitrarily from its friend set $F(x)$, and so $S(x) \subseteq F(x)$. At the end of such a term, $k + 1$ new surrogates are appointed to the corresponding large ancestor y of x from the friend set of y. We need to show that the friend set $F(y)$ of the appointing node y at this stage contains at least $k+1$ 10-friends of y of small degree, which can be appointed as its new surrogates. In fact, we will show something stronger, namely, that $F(y)$ contains at least $2k + 2$ (rather than $k + 1$) such friends. Hence $k + 1$ arbitrary points out of them will be appointed as the new surrogates of y, and the other at least $k + 1$ such friends may be used for compensating neighboring (due to cross edges) nodes.

At the outset all points are marked as *clean*. A point p remains clean as long as it is appointed as a surrogate of small nodes. However, at the first time p is appointed as a *new* surrogate of a large node it becomes *dirty*. Once p becomes dirty, it will never be re-appointed as a new surrogate again. Thus, each point can be appointed as a new surrogate of at most one large node, and this appointment is for a term of at least $\tau + 3 = \Omega(\tau)$ levels. We will make sure that during such a term the degree of any surrogate will increase by at most $\epsilon^{-O(d)} \cdot (2k + 1)$ units, and so the degree bound will be in check.

Our construction will guarantee that either all surrogates of a node are clean, or they are all dirty. We henceforth say that a node is *clean* (respectively, *dirty*) if all its surrogates are clean (resp., dirty). A dirty node will be complete, whereas a clean node may be complete or incomplete. Also, a large node will be dirty, whereas a small node is usually (except for a leech) clean; see Section 3.1.3 for more details.

3.1.3 A Symbiosis Between Leeches and Hosts. Appointing new surrogates to large nodes is a costly operation, and should be avoided whenever possible. Specifically, before new surrogates are

[6]The equation $D(p, i) = S(p, i) \subseteq F(p, i)$ holds for a small *non-leech* (p, i), but does not necessarily hold for a small *leech*. The definition of a leech (and non-leech) is given in Section 3.1.3.

appointed to a large node (p,i) (which happens at the beginning of each new term), we look for a dirty node (q,i) whose surrogates' term has not finished yet, which is a 24-friend of (p,i), i.e., $\delta(p,q) \leq 24 \cdot 5^i$. If no such node is found, we appoint $k+1$ new surrogates to (p,i) as described above. Otherwise, we re-use the surrogates of an arbitrary such 24-friend (q,i) of (p,i) as the surrogates of (p,i), i.e., we assign $S(p,i) = S(q,i)$. Observe that (q,i) is dirty and thus also complete; hence this assignment turns (p,i) too into a dirty and thus complete node. We say that (q,i) is a *leech* and (p,i) is its *host*.

As mentioned, it is advantageous for a node to have (at least) $k+1$ surrogates, or in other words, to be complete. We use the leech-host idea described above to turn clean (and possibly incomplete) nodes into dirty (and necessarily complete) ones. Specifically, recall that a clean node (p,i) re-uses all the surrogates of its children by appointing them as its surrogates. Before this is done, we look for a potential host for (p,i), i.e., a dirty non-leech 24-friend (q,i) of (p,i). If no such node is found, we assign $S(p,i) = D(p,i)$ as before. Otherwise, we re-use the surrogates of (q,i) as the surrogates of (p,i), i.e., we assign $S(p,i) = S(q,i)$, which turns (p,i) into a dirty and complete node.

A node may be the host of many leeches, whereas a leech has only one host. Moreover, a node cannot be both a leech and a host, and it may be neither of them. Being a leech is an inherited trait: If (p,i) is a leech of (q,i), then (p,i)'s parent is close enough (i.e., a 24-friend) to become a leech of (q,i)'s parent (this is irrelevant if the term of the host's surrogates is over). However, in case (p,i) has a dirty non-leech sibling whose surrogates' term is not over, we will prevent (p,i)'s parent from becoming a leech. In this case (p,i)'s parent will re-use the surrogates of such a dirty non-leech sibling of (p,i), and will become a dirty non-leech. This is the only scenario when we prevent a node from becoming a leech.

A clean node is necessarily small. On the other hand, a dirty node is not necessarily large. The only exception is when the dirty node is a leech. Indeed, a leech may be both small and dirty. On the other hand, we will show that a non-leech dirty node must be large (see Lemma 3.14). We will also show that for a clean node (p,i), we have $D(p,i) = S(p,i) \subseteq F(p,i)$. For a dirty node, this equation does not hold. We remark that a clean node may be complete or incomplete, and it may contain up to $2k+1$ surrogates. On the other hand, a dirty node must be complete, and it contains exactly $k+1$ surrogates.

There is an interesting interplay between leeches and hosts. On the negative side, a leech exploits the host by using its surrogates and overloading their degree due to cross edges that are incident on that leech. However, each host will have at most $O(1)^{O(d)}$ leeches at each level, and so the surrogates' degrees due to leeches will be greater than the surrogates' degrees due to the host by a small factor.
On the bright side, there are important advantages of exploiting the host. First, this allows us to "ignore" some cross edges of the basic spanner construction (and reduce the surrogates' degrees), specifically, those between leeches and their host as well as between all leeches of the same host; indeed, such cross edges connect nodes with the same surrogate set, and are thus *redundant*. Second, since leeches do not appoint new surrogates, the clean friends in $F(y)$ of the corresponding host y will remain clean until the term of its surrogates $S(y)$ is over. Consequently, this approach enables y and its non-leech ancestors to accumulate more and more clean points in their friend sets. Whenever a new term starts, new host and leeches are determined. At this stage we will have enough clean points in $F(\tilde{y})$ to appoint as surrogates of the new host $\tilde{y}$. Finally, we remark that an ancestor of a leech (respectively, host) can become a host (resp., leech). To summarize, there is a symbiosis between leeches and hosts, from which everyone enjoy.

3.1.4 Putting it All Together. Recall that once the surrogate sets $S(x)$ of all nodes x in T are computed, the FT spanner $\mathcal{H}$ is obtained by replacing each edge of the basic spanner H with a bipartite clique. We remark that *redundant edges* that connect nodes with the same surrogate set are disregarded.

We next show how to compute the surrogate sets of nodes, by putting all the ingredients that were described in Sections 3.1.1-3.1.3 together. The surrogate sets and the auxiliary sets (the descendant, friend and reserve sets) are computed bottom-up. That is, we first compute these sets for the 0-level nodes (i.e, the leaves of T), then for the 1-level nodes, and so forth. More specifically, we employ Procedure $\mathit{ComputeSets}_{(i)}$ to compute these sets for the i-level nodes, for $i = 0, \ldots, \ell$. Procedure $\mathit{ComputeSets}_{(i)}$ has two parts. The first part of this procedure computes the descendant, friend and reserve sets of the i-level nodes. (We omit the computation of the descendant sets of the i-level nodes in the first part of

Procedure $\mathit{ComputeSets}_{(i)}$, $i \in [0, \ell]$, as it can be done in a straightforward manner.) Equipped with these sets, the second part of Procedure $\mathit{ComputeSets}_{(i)}$ computes the surrogate sets of the i-level nodes.

There is a difference between Procedure $\mathit{ComputeSets}_{(0)}$ (which corresponds to the case $i = 0$) and Prcoedure $\mathit{ComputeSets}_{(i)}$, for $i \geq 1$. We start with describing Procedure $\mathit{ComputeSets}_{(0)}$.

The first part of Procedure $\mathit{ComputeSets}_{(0)}$ goes over the 0-level (leaf) nodes in an arbitrary order. Consider an arbitrary leaf node $(p, 0)$. We first put p in both the reserve set $R(p, 0)$ and the friend set $F(p, 0)$ of $(p, 0)$. Next, for every cross edge $((p, 0), (q, 0))$ that is incident on $(p, 0)$ in the basic spanner H (including redundant edges), we put q in the reserve set $R(p, 0)$ of $(p, 0)$. Note that q is a γ-friend of $(p, 0)$. If q is also a 10-friend of $(p, 0)$, we add it to the friend set $F(p, 0)$ of $(p, 0)$; we stop once $|F(p, 0)| = 3k+3$.

Having computed the friend and reserve sets $F(p, 0)$ and $R(p, 0)$ for all leaf nodes $(p, 0)$, the second part of Procedure $\mathit{ComputeSets}_{(0)}$ starts. As opposed to the first part of the procedure, here it is important to handle nodes $(p, 0)$ with $|F(p, 0)| \geq 2k + 2$ (if any) first, and only later handle the rest of the nodes. (See the first remark below to understand why this order is important.) Consider a leaf node $(p, 0)$. We first look for a potential host for $(p, 0)$, i.e., a dirty non-leech 24-friend $(q, 0)$ of $(p, 0)$.

- If such a node $(q, 0)$ is found, then we assign $S(p, 0) = S(q, 0)$, and $(p, 0)$ will be a leech of $(q, 0)$, and thus dirty and complete.
- Otherwise we check whether $|F(p, 0)| < 2k + 2$.
  - If so, $(p, 0)$ will be small and clean, and we assign $S(p, 0) = D(p, 0) = \{p\}$. (Notice that $S(p, 0) \subseteq F(p, 0)$, and so $|F(p, 0) \cup S(p, 0)| < 2k + 2$, and $(p, 0)$ is small by definition.)
  - In the complementary case (i.e., $|F(p, 0)| \geq 2k + 2$) $(p, 0)$ will be large; in this case we appoint $k + 1$ points from $F(p, 0)$ as *new surrogates* of $(p, 0)$ for a term of $\Omega(\tau)$ levels (see the second remark below), and these points become dirty. Also, $(p, 0)$ itself will be dirty and complete.

We remind that at the outset all points are marked as clean. A point p becomes dirty when it is appointed as a new surrogate of a large node; p will not be re-appointed as a new surrogate again. We say that a node is clean (respectively, dirty) if all its surrogates are clean (resp., dirty).

Next, we describe Procedure $\mathit{ComputeSets}_{(i)}$, for $i \geq 1$, which computes the surrogate set $S(p, i)$, the descendant set $D(p, i)$, the friend set $F(p, i)$ and the reserve set $R(p, i)$, for all i-level nodes (p, i). We invoke Procedure $\mathit{ComputeSets}_{(i)}$ only after all Procedures $\mathit{ComputeSets}_{(0)}, \ldots, \mathit{ComputeSets}_{(i-1)}$ have terminated. At this stage, the surrogate set $S(q, j)$, the descendant set $D(q, j)$, the friend set $F(q, j)$ and the reserve set $R(q, j)$ have been computed, for all nodes (q, j) at levels $j \in [0, i - 1]$.

The first part of Procedure $\mathit{ComputeSets}_{(i)}$ goes over the i-level nodes in an arbitrary order. Consider an arbitrary i-level node (p, i). We first put in $R(p, i)$ (respectively, $F(p, i)$) all *clean* points of $R(q, i - 1)$ (resp., $F(q, i - 1)$), for every child $(q, i - 1)$ of (p, i). In addition, all (clean) points of $R(p, i)$ that are 10-friends of (p, i) but do not belong to $F(p, i)$ are added to $F(p, i)$; we stop once $|F(p, i)| = 3k + 3$. This concludes the computation of $F(p, i)$, but the computation of $R(p, i)$ is not over yet. We pause the computation of $R(p, i)$ for now, and proceed to computing the friend sets $F(x)$ of all i-level nodes in this way. Having computed the friend sets of all i-level nodes, we return to completing the computation of $R(p, i)$. For every cross edge $((p, i), (q, i))$ that is incident on (p, i) in the basic spanner H (including redundant edges), we add to the reserve set $R(p, i)$ of (p, i) all (clean) points from the friend set $F(q, i)$ of (q, i) as well as all (clean) 10-friends of (q, i) from the reserve set $R(q, i)$ of (q, i), disregarding points that were already in $R(p, i)$. This concludes the computation of $R(p, i)$.
Observe that any 10-friend of a child $(q, i-1)$ of (p, i) is also a 10-friend of (p, i) (see Claim 3.1 in Section 3.2.1). Hence, by construction, all points of $F(p, i)$ are 10-friends of (p, i). Similarly, any $(\gamma + 10)$-friend of a child $(q, i - 1)$ of (p, i) is also a $(\gamma + 10)$-friend (in fact, a γ-friend) of (p, i). Also, recall that the weight of i-level cross edges in the basic spanner H is at most $\gamma \cdot 5^i$. Hence, by construction, all points of $R(p, i)$ are $(\gamma + 10)$-friends of (p, i).

Having computed the friend and reserve sets $F(p, i)$ and $R(p, i)$ for all i-level nodes (p, i), the second part of Procedure $\mathit{ComputeSets}_{(i)}$ starts. As opposed to the first part of the procedure, here it is important to handle nodes (p, i) with $|F(p, i)| \geq 2k + 2$ first, and only later handle the rest of the nodes (see the first remark below). Consider a node (p, i). The execution of the procedure splits into two cases. *Case 1: All children of* (p, i) *are clean.* We first look for a potential host for (p, i), i.e., a dirty non-leech 24-friend (q, i) of (p, i).

- If such a node (q, i) is found, then we assign $S(p, i) = S(q, i)$, and (p, i) will be a leech of (q, i), and thus dirty and complete.
- Otherwise we check whether $|F(p, i)| < 2k + 2$.
  - If so, (p, i) will be small and clean, and we assign $S(p, i) = D(p, i)$. (We will show in Corollary 3.8 that $S(p, i) \subseteq F(p, i)$, and so $|F(p, i) \cup S(p, i)| < 2k + 2$, and (p, i) is small by definition.)
  - In the complementary case (i.e., $|F(p, i)| \geq 2k + 2$) (p, i) will be large; in this case we appoint $k + 1$ points from $F(p, i)$ as *new surrogates* of (p, i) for a term of $\Omega(\tau)$ levels (see the second remark below), and these points become dirty. Also, (p, i) itself will be dirty and complete.

Case 2: At least one child of (p, i) *is dirty.* We first look for the dirty non-leech child $(q, i - 1)$ of (p, i) (if any) whose surrogates' term started most recently. (In fact, our analysis shows that any dirty non-leech child of (p, i) whose surrogates' term is not over at level $i - 1$ will do. However, it is more instructive and natural to choose the one whose surrogates' term started most recently.)

- If such a child $(q, i - 1)$ is found and the term of its surrogates is not over at level $i - 1$, we assign $S(p, i) = S(q, i - 1)$. In this case (p, i) becomes a dirty and complete non-leech.
- Otherwise, the term of the surrogates of each dirty non-leech child of $(q, i - 1)$ (if any) will be over at level $i - 1$. In this case we proceed similarly to case 1. Specifically, we first look for a potential host (q, i) for (p, i), i.e., a dirty non-leech 24-friend (q, i) of (p, i).
  - If such a node (q, i) is found, then we assign $S(p, i) = S(q, i)$, and (p, i) will be a leech of (q, i), and thus dirty and complete.
  - Otherwise, we appoint $k + 1$ points from $F(p, i)$ as *new surrogates* of (p, i) for a term of $\Omega(\tau)$ levels (see the second remark below), and these points become dirty. Also, (p, i) itself will be dirty and complete. Unlike case 1 where we may have $|F(p, i)| < 2k + 2$ and (p, i) will be clean, in this case it must hold that $|F(p, i)| \geq 2k + 2$, implying that (p, i) is large, and thus dirty and complete. To this end, we will show (see Corollary 3.15) that whenever the need to appoint new surrogates for an ancestor (p, i) of a dirty node arises, we have $|F(p, i)| \geq 2k + 2$.

Remarks:

1. The second part of Procedure $\mathit{ComputeSets}_{(i)}$ (for any i, including $i = 0$) handles nodes (p, i) with $|F(p, i)| \geq 2k + 2$ (if any) first, and only later handles the rest of the nodes. The reason we handle nodes in this order is that when looking for a potential host (q, i) for (p, i), we are in fact looking for a dirty non-leech 24-friend of (p, i). However, we do not want (q, i) to become dirty after handling (p, i), assuming (p, i) is clean, thus missing a potential host for (p, i). Handling nodes in this order guarantees that all the 24-friends of a dirty non-leech node must be dirty as well.

2. We mentioned that new surrogates of large nodes are appointed for a term of $\Omega(\tau)$ levels. The term of surrogates consists of two phases. The first phase starts with their appointment, and ends when their degree due to non-redundant cross edges (i.e., disregarding cross edges that connect nodes with the same surrogate set) *since their appointment* reaches $k + 1$. Notice that the degree of a surrogate before its appointment at a large node may be positive, but we only consider its degree since this appointment. Moreover, observe that the degree of a surrogate may also increase due to cross edges that are NOT incident on the relevant host but rather on the leeches of this host.

The first phase may end within a single level, but may also last for many levels. When the second phase starts, the degree of surrogates (since their appointment) is between $k+1$ and $k+\xi^2(2k+1)$. The second phase lasts precisely $\tau+2$ levels. Thus any term seems to last at least $\tau+3$ levels. However, this is true only if these surrogates are re-used over and over, which need not be the case in general. Recall that the computation of $S(p,i)$ in Case 2 above started by looking for the dirty non-leech child $(q,i-1)$ of (p,i) whose surrogates' term started most recently, and setting $S(p,i)=S(q,i-1)$ if such a child is found. Consider another non-leech child $(r,i-1)$ of (p,i). Setting $S(p,i)=S(q,i-1)$ forces the term of points in $S(r,i-1)$ to be over at level $i-1$. (We will show in Claim 3.10 that the surrogate sets of dirty non-leech nodes at the same level are pairwise disjoint, and so $S(q,i-1)\cap S(r,i-1)=\emptyset$.) But this is fine, because the surrogates of $S(p,i)=S(q,i-1)$ that (p,i) chose to re-use are already sufficient for our needs, and so we can safely disregard the surrogates of all other dirty non-leech children of (p,i). In particular, this approach guarantees that once a large i-level node x appoints to itself $k+1$ new surrogates, no ancestor of x at level at most $i+\tau+2$ will appoint to itself new surrogates.

3.2 Degree Analysis

In this section we analyze the degree of the spanner construction $\mathcal{H}$ described in Section 3.1.

3.2.1 Getting Started. We start with some basic definitions and claims. For a node $x=(p,i)$, recall that $p(x)$ denotes the corresponding net-point p. The distance between two nodes x and y is defined as the distance between their net-points $p(x)$ and $p(y)$, i.e., $\delta(x,y)=\delta(p(x),p(y))$. The distance between a node x and a point p can be defined similarly, i.e., $\delta(x,p)=\delta(p(x),p)$. Recall that an i-level node x is called a *t-friend* of another i-level node y (respectively, of a point p) if $\delta(x,y)\le t\cdot 5^i$ (resp., $\delta(x,p)\le t\cdot 5^i$). For a point p and any $i\in[0,\ell]$, the *(i-level) base bag* of p is the unique i-level node x such that $p\in D(x)$.

Claim 3.1 (Once a Friend, Always a Friend) *(1) If a point p is a 35-friend (let alone a 10-friend) of an i-level node x, then it is also a 10-friend of every ancestor of x. (2) If a point p is a $(\gamma+68)$-friend (let alone a γ-friend) of x, then it is also a γ-friend of every ancestor of x.*

Proof: Suppose first that p is a 35-friend of x. To prove the first assertion, it suffices to show that p is a 10-friend of the parent $\pi(x)$ of x. Indeed, we have

$$\delta(\pi(x),p) \;\le\; \delta(\pi(x),x)+\delta(x,p) \;\le\; 3\cdot 5^{i+1}+35\cdot 5^i \;=\; 10\cdot 5^{i+1}.$$

To prove the second assertion, suppose that p is a $(\gamma+68)$-friend of x. Assuming $\gamma\ge 21$, we have

$$\delta(\pi(x),p) \;\le\; \delta(\pi(x),x)+\delta(x,p) \;\le\; 3\cdot 5^{i+1}+(\gamma+68)\cdot 5^i \;<\; \gamma\cdot 5^{i+1}. \qquad \blacksquare$$

Claim 3.2 (Being a Leech is an Inherited Trait) *If an i-level node y is a leech of some i-level node x, then y's parent $\pi(y)$ is a 24-friend of x's parent $\pi(x)$, and is thus close enough to become its leech.*

Proof: If y is a leech of x, then the net-point $p(y)$ of y is a 24-friend of x. The first assertion of Claim 3.1 implies that $p(y)$ is a 10-friend of $\pi(x)$, i.e., $\delta(p(y),\pi(x))\le 10\cdot 5^{i+1}$. Since $\delta(\pi(y),y)=\delta(\pi(y),p(y))\le 3\cdot 5^{i+1}$, it follows that

$$\delta(\pi(y),\pi(x)) \;\le\; \delta(\pi(y),p(y))+\delta(p(y),\pi(x)) \;\le\; 3\cdot 5^{i+1}+10\cdot 5^{i+1} \;<\; 24\cdot 5^{i+1}. \qquad \blacksquare$$

Claim 3.3 (Distant Friends Soon Become Close Friends) *(1) For any i-level cross edge (x,y) of the basic spanner H and any $p\in S(x), q\in S(y)$, $\delta(p,q)\le(\gamma+68)5^i$. In other words, for any i-level cross edge (p,q) in the FT spanner $\mathcal{H}$, $\delta(p,q)\le(\gamma+68)5^i$. (2) Let p be a 10-friend of some i-level node x. Any point q for which $\delta(p,q)\le(\gamma+68)5^i$ is a 10-friend of the $(i+\tau)$-level ancestor of x.*

Proof: The weight of an i-level cross edge (x,y) in H is at most $\gamma 5^i$. For any $p \in S(x)$, we have $\delta(x,p) \leq 34 \cdot 5^i$; refer to Observation 3.9 and the paragraph preceding it to see why this upper bound holds. Similarly, we have $\delta(y,q) \leq 34 \cdot 5^i$. Consequently, $\delta(p,q) \leq \delta(p,x) + \delta(x,y) + \delta(y,q) \leq (\gamma + 68)5^i$.

Next, we prove the second assertion. Let x' be the $(i+\tau)$-level ancestor of x, and notice that $\delta(x',x) \leq 4 \cdot 5^{i+\tau}$. We have $\delta(x',q) \leq \delta(x',x) + \delta(x,p) + \delta(p,q) \leq 4 \cdot 5^{i+\tau} + 10 \cdot 5^i + (\gamma+68)5^i \leq 10 \cdot 5^{i+\tau}$, where the last inequality holds for $\gamma \geq 3$. It follows that q is a 10-friend of x', and we are done. ▮

By construction, a node with at least one dirty child must be dirty too. (See Case 2 in Procedure $ComputeSets_{(i)}$, $i \geq 1$.) This means that all ancestors of a dirty node are dirty, or equivalently, all descendants of a clean node are clean. Notice also that the construction guarantees that any dirty node must be complete. More specifically, a dirty node has exactly $k+1$ surrogates. We summarize these observations in the following statement.

Observation 3.4 (Once Dirty, Always Dirty and Complete) *All ancestors of a dirty node (including itself) are dirty. Also, each dirty node has exactly $k+1$ surrogates, and is thus complete.*

Remark: Note that any large node is dirty. Even though the converse statement holds for non-leech nodes (see Lemma 3.14 in Section 3.2.2), it does not necessarily hold for leeches. Nevertheless, it can be proved that all ancestors of a (possibly leech) dirty node are in fact large. (This assertion is stronger than Observation 3.4 and requires proof; we do not provide the proof of this assertion since we do not use it in the sequel.)

The following observation is implied by Observation 3.4 and the construction.

Observation 3.5 (The Descendants of Clean Nodes are Surrogates) *For any clean node x, we have $S(x) = D(x)$. Since any descendant x' of x is clean, it follows that $S(x') \subseteq S(x)$.*

Remark: Let p be an arbitrary point, and let $i \in [0, \ell]$. Observation 3.5 shows that the only clean i-level node x (if any) of which p is a surrogate (i.e., $p \in S(x)$) is the i-level base bag of p. Consequently, if the i-level base bag of p is dirty, then p will not be a surrogate of j-level clean nodes, for any $j \geq i$.

The next lemma shows that all descendants (and thus all surrogates) of a clean node must be clean.

Claim 3.6 (All Descendant Points of Clean Nodes are Clean) *For a clean j-level node x, all points of $D(x)$ are clean at level j. In other words, if a point p is dirty at level j, then the j-level base bag of p is dirty too.*

Proof: Suppose for contradiction that there is a point $p \in D(x)$ that is dirty at level j, and let z be the appointing node which made p dirty. Observe that z is a dirty non-leech i-level node, such that $p \in S(z)$ and $i \leq j$. Let $\tilde{x}$ be the i-level base bag of p, i.e., $p \in D(\tilde{x})$. Note that x is the j-level base bag of p, and so $\tilde{x}$ is a descendant of x. By Observation 3.4, $\tilde{x}$ must be clean. Hence $\tilde{x} \neq z$. Note also that p is a 4-friend of $\tilde{x}$ and a 10-friend of z. It follows that $\tilde{x}$ is a 14-friend of z, and would become its leech and thus dirty, yielding a contradiction. ▮

We have shown in Claim 3.6 that all surrogates of a clean node are clean. Also, all surrogates of a dirty node are dirty by the construction. We summarize this property in the following statement.

Corollary 3.7 *All surrogates of a clean (respectively, dirty) node are clean (resp., dirty).*

The following corollary follows from Observation 3.5, Claim 3.6 and the construction.

Corollary 3.8 *For a clean node x, we have $S(x) = D(x) \subseteq F(x)$, and so $|S(x)| = |D(x)| \leq |F(x)| = |F(x) \cup S(x)| < 2k+2$.*

Observation 3.5 implies that all surrogates of a clean node are 4-friends of this node. The surrogates of a dirty non-leech node are 10-friends of it, but the surrogates of a (dirty) leech are not necessarily 10-friends of it. Since a leech is a 24-friend of its host, it follows that the surrogates of a leech are 34-friends of it. We summarize these observations in the following statement.

Observation 3.9 *All surrogates of either a clean or dirty node are 34-friends of it.*

Recall that our construction makes a persistent effort to re-use old surrogates. In particular, if an i-level node x has a dirty non-leech child y, such that the term of the $k+1$ surrogates of $S(y)$ is not over at level $i-1$, then we set $S(x) = S(y)$. (If x has other such children, then it may choose to use surrogates from another child.) In this case x will be a dirty non-leech node as well, and we say that x and y are *copies*. Note that y itself may be a copy of one of its dirty non-leech children, and so forth. In general, there is a path of dirty non-leech copies in the tree which leads from a dirty non-leech node x down to its *appointing copy* $a(x)$, which is a dirty large node which appoints $k+1$ new surrogates. By construction, all these non-leech copies are dirty and complete; in fact, we will show in Claim 3.13 that they are large.

Recall that surrogates of large nodes are appointed for a term of at least $\tau + 3$ levels. This does not mean that a specific surrogate will necessarily be re-used for $\Omega(\tau)$ levels due to an abundance of surrogates; see the second remark at the end of Section 3.1.4. This only means that no j-level ancestor of an i-level appointing node, for any $j \le i + \tau + 2$, will appoint to itself new surrogates. We define $term(x)$ as the term between the level of the appointing copy $a(x)$ of x and the highest level of any copy of x (which is either x or an ancestor of x). By construction, $term(x)$ lasts at least $\tau + 3$ levels.

The "disjointness" properties of Claims 3.10 and 3.11 will be heavily used in the sequel.

Claim 3.10 (Surrogates and Old Friends are Disjoint) *For any pair x, y of (distinct) dirty non-leeches at the same level, $S(x) \cap S(y) = F(a(x)) \cap F(a(y)) = \emptyset$. More generally, let $F^*(a(x))$ and $F^*(a(y))$ denote the set of all 10-friends of $a(x)$ and $a(y)$, respectively. Then $F^*(a(x)) \cap F^*(a(y)) = \emptyset$.*

Proof: Write $a(x) = (p, i)$ and $a(y) = (q, j)$, and suppose without loss of generality that the appointment of $S(a(x)) = S(x)$ takes place before the appointment of $S(a(y)) = S(y)$. It must hold that $i \le j$. Let x' be the j-level copy of x, with $S(x') = S(x)$. By construction, x' is a dirty non-leech.

Suppose for contradiction that there is a point s in $F^*(a(x)) \cap F^*(a(y))$. Hence s is a 10-friend of both $a(x)$ and $a(y)$. By Claim 3.1, s is also a 10-friend of x', and so x' and $a(y)$ are 20-friends. It follows that $a(y)$ would become a leech of x', and would not be an appointing node, a contradiction. ▮

Claim 3.11 (Dirty Surrogate Sets are Either Equal or Disjoint) *Let x be a dirty i-level node. For each level $j \in [i, \ell]$ and any j-level node y, either $S(x) = S(y)$ or $S(x) \cap S(y) = \emptyset$ must hold. In the former case $S(x) = S(y)$, y must be dirty.*

Proof: Let y be an arbitrary j-level node. If $S(x) \cap S(y) = \emptyset$, then we are done.

We henceforth assume that $S(x) \cap S(y) \neq \emptyset$. We will show that $S(x) = S(y)$, and that y is dirty.

Define v as x if it is a non-leech, and as the host of x otherwise. We have $S(x) = S(v)$.

We argue that $term(v)$ cannot be over until level $j-1$. Indeed, otherwise no point of $S(x)$ will be a surrogate of any j-level node, including y. It follows that $S(x) \cap S(y) = \emptyset$, a contradiction.

We henceforth assume that $term(v)$ is not over before level j. Consider the j-level copy v' of v, where $S(v') = S(v) = S(x)$. Observe that v' is a dirty non-leech. Note also that $S(v') \cap S(y) \neq \emptyset$, and let p be a point in $S(v') \cap S(y)$. Since v' is dirty, p must be dirty too. Claim 3.6 implies that y is dirty as well.

If $y = v'$, then $S(x) = S(y)$ must hold, and we are done. We henceforth assume that $y \neq v'$. In this case y must be a leech of v'. Indeed, otherwise we have $S(v') \cap S(y) = \emptyset$ by Claim 3.10, which is a contradiction. Since y is a leech of v', we have $S(y) = S(v') = S(x)$. The corollary follows. ▮

Claim 3.12 (Clean 10-Friends of Non-Leeches Remain Clean) *Let x be a dirty non-leech i-level node. Then all points in $F(x) \setminus S(x)$ remain clean during the entire $term(x)$. More generally, all the 10-friends of x that are clean at the beginning of level i will remain clean during the entire $term(x)$, except for the surrogates $S(x)$ of x which get dirty in the case when $x = a(x)$ is an appointing node.*

Proof: Let p be an arbitrary 10-friend of x that is clean at the beginning of level i. Suppose for contradiction that p became dirty at level i, and $p \notin S(x)$. This means that some other i-level appointing node y, $y \neq x$, appointed p as one of its $k+1$ new surrogates at level i. Since p is a 10-friend of both x and y, it follows that x and y are 20-friends. If x is either a non-appointing node (i.e., $x \neq a(x)$) or it is an appointing node which appoints its surrogates before y does, then y would become x's leech, and would not become an appointing node, a contradiction. Otherwise, x is a an appointing node which appoints its surrogates after y does, but then x would become y's leech, yielding the same contradiction.

We have shown that all 10-friends of x (except for those in $S(x)$) that are clean at the beginning of level i will remain clean at the end of level i, and so they will be clean at the beginning of level $i+1$. By Claim 3.3, all these points are 10-friends of $\pi(x)$, which is also a non-leech. Thus we can apply the same argument for level $i+1$, and carry on in this way also for subsequent levels. Consequently, we get that all 10-friends of x that are clean at the beginning of level i will remain clean during the entire $term(x)$, except for the surrogates $S(x)$ of x which get dirty in the case $x = a(x)$. ▮

Claim 3.13 (If the Appointing Copy is Large, All Other Copies are Large Too) *Let x be a dirty non-leech i-level node. Then there is a path of dirty non-leech copies of x in the tree which leads down from x to its appointing copy $a(x)$. Also, all points of $F(a(x)) \setminus S(x)$ belong to $F(x)$, unless $|F(x)| = 3k+3$. In particular, if $a(x)$ is large, then x will be large too.*

Proof: First, the fact that there is a path of dirty non-leech copies of x in the tree which leads down from x to its appointing copy $a(x)$ follows easily from the construction.

By Claim 3.12, all 10-friends of $a(x)$ that are clean after the appointment of $S(x) = S(a(x))$ (in particular, all points in $F(a(x)) \setminus S(a(x))$) remain clean during the entire $term(x)$. Hence they will be clean at level i. By construction, all the points in $F(a(x)) \setminus S(x)$ will belong to $F(x)$, unless $|F(x)| = 3k+3$.

If $a(x)$ is large, then we have $|F(a(x)) \setminus S(x)| \geq k+1$. Consequently, all the at least $k+1$ points in $F(a(x)) \setminus S(x)$ will belong to $F(x)$, unless $|F(x)| = 3k+3$. It follows that $|F(x) \cup S(x)| \geq 2k+2$, and so x is large. ▮

3.2.2 A Key Lemma and Its Corollary.

The following lemma is central in our analysis. In particular, it shows that whenever the need to appoint new surrogates for some node x arises, we have $|F(x)| \geq 2k+2$. (See Corollary 3.15.) Thus Procedure $ComputeSets_{(i)}$ from Section 3.1.4 is well-defined.

Lemma 3.14 (All Non-Leeches are Large) *Let x be a dirty non-leech i-level node. Then:*
(a) The appointing copy $a(x)$ of x is large, thus x is large too by Claim 3.13. (It is possible that $x = a(x)$.)
(b) If i is the last level of $term(x)$, then $|F(x)| \geq 2k+2$. (Note that $F(x) = F(x) \setminus S(x)$ in this case.) Moreover, if $|F(x)| < 3k+3$, then at least $k+1$ points from $F(x)$ do not belong to $F(a(x))$.

Proof: The proof of the two assertions of the lemma is by induction on i.

A dirty node that has only clean children is called *atomically-dirty*; otherwise it is *compound-dirty.* For the basis of the induction we may consider nodes whose appointing copy is atomically-dirty. Such nodes must be large by the construction. (An atomically-dirty non-leech y may get dirty only if $|F(y)| \geq 2k+2$, in which case y is large by definition.) The first assertion follows immediately.
To prove the second assertion we use an argument that works for both the basis of the induction and the induction step. We omit this argument here for conciseness, but provide it in the induction step.
Induction Step: Assume that the lemma holds for all smaller values of $i, i \geq 1$, and prove it for i.

We start with the first assertion. We have shown that the case when $a(x)$ is atomically-dirty is trivial.

We henceforth assume that $a(x)$ is compound-dirty. By definition, $a(x)$ has at least one dirty child. Let i_a be the level of $a(x)$, with $i_a \leq i$. Next, we argue that every dirty child of $a(x)$ is either a non-leech whose term is over at level $i_a - 1$, or a leech of some host whose term is over at level $i_a - 1$. Indeed, if $a(x)$ had a dirty non-leech child y whose term is not over at level $i_a - 1$, then Procedure $ComputeSets_{(i_a)}$ would assign $S(a(x)) = S(y)$, and $a(x)$ would not be an appointing copy, a contradiction. Similarly, if $a(x)$ had a dirty child which is a leech of some host y whose term is not over at level $i_a - 1$, then y's parent would be a non-leech by the construction. However, it is easy to see that $a(x)$ would be close enough to y's parent to become its leech by Claim 3.2, and would not be an appointing copy, a contradiction.

Consider such a node y of level $i_a - 1 < i$, which is either a dirty non-leech child of $a(x)$ or a (dirty) host of a leech child of $a(x)$. We have shown that $i_a - 1$ must be the last level of $term(y)$. By the induction hypothesis for y, $|F(y)| \geq 2k + 2$. By construction, all points of $F(y)$ are clean at the beginning of level $i_a - 1$. By Claim 3.12, all points of $F(y)$ will remain clean at level $i_a - 1$, and thus will be clean at the beginning of level i_a. If y is a child of $a(x)$, then for any point s of $F(y)$, $\delta(a(x), s) \;\leq\; \delta(a(x), y) + \delta(y, s) \;\leq\; 3 \cdot 5^{i_a} + 10 \cdot 5^{i_a - 1} \;\leq\; 5 \cdot 5^{i_a}$. Otherwise, y is a host of a child c of $a(x)$, in which case we have

$$\delta(a(x), s) \;\leq\; \delta(a(x), c) + \delta(c, y) + \delta(y, s) \;\leq\; 3 \cdot 5^{i_a} + 24 \cdot 5^{i_a - 1} + 10 \cdot 5^{i_a - 1} \;\leq\; 10 \cdot 5^{i_a}.$$

In both cases $a(x)$ has at least $2k + 2$ clean 10-friends at the beginning of level i_a, namely, the points of $F(y)$. By construction, all these points will belong to $F(a(x))$, unless $|F(a(x))| = 3k + 3$. In either case we have $|F(a(x))| \geq 2k + 2$, implying that $a(x)$ is large. The first assertion follows.

Next, we prove the second assertion. We henceforth assume that i is the last level of $term(x)$. If $|F(x)| = 3k+3$, then we are done. We henceforth assume that $|F(x)| < 3k+3$. The first assertion implies that $a(x)$ is large, and so there must be at least $k + 1$ clean 10-friends of $a(x)$ in $F(a(x)) \setminus S(a(x))$. By Claim 3.13, all these points remain clean during the entire $term(x)$, and they will belong to $F(x)$.

Consider the last level of the first phase in $term(x)$, denoted l, and let $q_1, \ldots, q_{k+1}$ be $k + 1$ arbitrary points (among a total of at most $k + \xi^2 \cdot (2k+1)$ points) which increased the degree of points in $S(a(x)) = S(x)$ since the beginning of $term(x)$ (due to non-redundant cross edges). For each point q_j, let l_j be the first level in $term(x)$, such that the basic spanner H contains a cross edge between either the l_j-level copy x_j of x or one of its leeches and some l_j-level node y_j for which both $q_j \in S(y_j)$ and $S(y_j) \neq S(x)$ hold; observe that $l_j \leq l$. (For such a *non-redundant* edge, our FT spanner $\mathcal{H}$ contains a bipartite clique between $S(x)$ and $S(y_j)$.) By Claim 3.11, $S(y_j) \cap S(x) = \emptyset$. Moreover, we argue that q_j cannot be a 10-friend of $a(x)$, which implies that it does not belong to $F(a(x))$. If y_j is a dirty non-leech, this assertion follows immediately from Claim 3.10. In the case that y_j is a dirty leech, we can apply Claim 3.10 with the host of y_j instead of y_j itself, and get the same result. We henceforth assume that y_j is clean. Since $q_j \in S(y_j)$, Observation 3.5 implies that q_j is a 4-friend of y_j. If q_j were a 10-friend of $a(x)$, then it must also be a 10-friend of the l_j-level copy x_j of x. Consequently, y_j would be a 14-friend of x_j, and would thus become its leech and dirty, a contradiction.

The second phase of $term(x)$ starts at level $l + 1$, and ends at level $i = l + \tau + 2$. Suppose first that all $k + 1$ points $q_1, \ldots, q_{k+1}$ are clean at the beginning of level $i - 2 = l + \tau$. In this case the l_j-level node y_j for which $q_j \in S(y_j)$ must be clean by Corollary 3.7 and the construction, for each $j \in [k + 1]$. Moreover, by Claim 3.3, all the $k + 1$ points $q_1, \ldots, q_{k+1}$ are 10-friends of the $(i - 2)$-level copy of x. Claim 3.1 implies that they will also be 10-friends of all the ancestors of that node. Moreover, all these points will remain clean until level i by Claim 3.12. Finally, recall that all $k + 1$ points $q_1, \ldots, q_{k+1}$ are not 10-friends of $a(x)$, and thus they do not belong $F(a(x))$. On the other hand, we argue that these points will belong to $F(x)$ by the construction. Indeed, since there is a cross edge between y_j and either the l_j-level copy x_j of x or one of its leeches, for each $j \in [k + 1]$, it follows that q_j will belong to the reserve set of either x_j or one of its leeches. Consequently, q_j will also belong to the reserve set of the $(i - 2)$-level copy x_{i-2} of x. (The latter assertion is immediate if the cross edge is between y_j and x_j. In the complementary case where the cross edge is between y_j and some leech of x_j, q_j will belong to the reserve set of the $(i - 2)$-level ancestor of that leech, denoted w_{i-2}. If $w_{i-2} = x_{i-2}$, then we are done. Otherwise, there is a cross edge between w_{i-2} and x_{i-2}. Claim 3.3 implies that q_j is a 10-friend of w_{i-2}, hence our construction guarantees that q_j will be added to the reserve set of x_{i-2}.) Since q_j is a 10-friend of x and $|F(x)| < 3k + 3$, it follows that q_j will belong to $F(x)$. We conclude that all points $q_1, \ldots, q_{k+1}$ belong to $F(x)$, which completes the proof of the second assertion in this case.

We henceforth assume that some point q_j became dirty until level $i - 3$. Recall that l_j is the first level in $term(x)$, such that H contains a non-redundant edge between either the l_j-level copy x_j of x or one of its leeches and some node y_j for which both $q_j \in S(y_j)$ and $S(y_j) \cap S(x) = \emptyset$ hold. Let l'_j be the first level after l_j (i.e., $l'_j > l_j$) in which the l'_j-level copy x'_j of x and the l'_j-level ancestor y'_j of y_j are 24-friends. We argue that $l'_j \leq i-2$. Indeed, as mentioned above, Claim 3.3 implies that q_j is a 10-friend of the $(i-2)$-level copy of x. Also, by Observation 3.9, the fact that q_j is in $S(y_j)$ implies that it is a 34-friend of y_j. By Claim 3.1, q_j is a 10-friend of any ancestor of y_j. It follows that the $(i-2)$-level copy of x and the $(i-2)$-level ancestor of y_j are 20-friends, and so $l'_j \leq i-2$.

Consider now level $\tilde{l}_j = l'_j - 1$, where $l_j \leq \tilde{l}_j \leq i-3$, and let $\tilde{x}_j$ and $\tilde{y}_j$ be the $\tilde{l}_j$-level ancestors of x_j and y_j, respectively. We argue that $S(\tilde{y}_j) \cap S(x) = \emptyset$. Indeed, this assertion clearly holds if $\tilde{l}_j = l_j$. Consider the complementary case $\tilde{l}_j > l_j$, and suppose for contradiction that $S(\tilde{y}_j) \cap S(x) \neq \emptyset$. Since $S(x) = S(\tilde{x}_j)$, Claim 3.11 implies that $S(\tilde{y}_j) = S(\tilde{x}_j)$ must hold. However, in this case $\tilde{y}_j$ must be a leech of $\tilde{x}_j$ and thus a 24-friend of it, which stands in contradiction to the above definitions.

The analysis splits into two main cases.

Case 1: $\tilde{y}_j$ is a clean node. We argue that q_j must become dirty at level $l'_j = \tilde{l}_j + 1$ (not before and not after). To see this, first note that q_j cannot become dirty at any level between l_j and $\tilde{l}_j$. Indeed, otherwise the appointing node that made q_j dirty would be a 20-friend of the corresponding ancestor of y_j, which would become its leech and thus dirty. However, then $\tilde{y}_j$ would be dirty too by Observation 3.4, a contradiction. On the other hand, if q_j does not get dirty at level l'_j or before, then it must get dirty afterwards (because it gets dirty until level $i-3$). Suppose for contradiction that some node z at level h appoints q_j as one of its $k+1$ new surrogates, where $l'_j + 1 \leq h \leq i-3$. This means that $\delta(z, q_j) \leq 10 \cdot 5^h$. Recall that q_j is a 10-friend of any ancestor of y_j, and in particular of y'_j. Also, y'_j is a 24-friend of x'_j, and so $\delta(x'_j, q_j) \leq \delta(x'_j, y'_j) + \delta(y'_j, q_j) \leq 24 \cdot 5^{l'_j} + 10 \cdot 5^{l'_j} < 7 \cdot 5^{l'_j+1} \leq 7 \cdot 5^h$. (The last inequality holds since $h \geq l'_j + 1$.) Let x_h be the h-level copy of x. Observe that $\delta(x_h, q_j) \leq \delta(x_h, x'_j) + \delta(x'_j, q_j) \leq 4 \cdot 5^h + 7 \cdot 5^h = 11 \cdot 5^h$. Consequently, $\delta(x_h, z) \leq \delta(x_h, q_j) + \delta(q_j, z) \leq 11 \cdot 5^h + 10 \cdot 5^h \leq 21 \cdot 5^h$. Thus x_h and z are 21-friends, and so z would have to become a leech of x_h, and would not be an appointing node, a contradiction. It follows that q_j becomes dirty at level $l'_j = \tilde{l}_j + 1$, as was argued above.

Denote by u the l'_j-level appointing node that appoints q_j as one of its $k+1$ new surrogates at level l'_j. By construction, u is a dirty non-leech. By the induction hypothesis, u must be large. Thus there are at least $k+1$ clean 10-friends of u in $F(u) \setminus S(u)$. Claim 3.10 implies that $F(u) \cap F(a(x)) = \emptyset$. By Claim 3.13, all the at least $k+1$ points in $F(u) \setminus S(u)$ remain clean during the entire *term*(u). Moreover, recall that y'_j is a 24-friend of x'_j and a 10-friend of q_j, and so x'_j and q_j are 34-friends. Since u is a 10-friend of q_j, we get that x'_j and u are 44-friends. Hence there is a cross edge between x'_j and u (assuming $\gamma > 44$). Since u is dirty, it follows that the second phase of *term*(x) will start at level $l'_j + 1$ (or before), which implies that *term*(u) will last at least until the last level i of *term*(x). In particular, all the at least $k+1$ points in $F(u) \setminus S(u)$ will remain clean until level i. Since x'_j and u are 44-friends, all points of $F(u) \setminus S(u)$ are 54-friends of x'_j. All these points are also 10-friends of the $(l'_j + 2)$-level copy of x, and let alone of x itself. By construction, since there is a cross edge between x'_j and u (and as $|F(x)| < 3k+3$), all points of $F(u) \setminus S(u)$ will belong to $F(x)$. Hence, at least $k+1$ points from $F(x)$ do not belong to $F(a(x))$, and we have $|F(x)| \geq |F(a(x)) \setminus S(a(x))| + |F(u) \setminus S(u)| \geq 2k+2$, which provides the required result.

Case 2: $\tilde{y}_j$ is a dirty node. It is possible that $\tilde{y}_j$ is a leech; in this case we consider the host of $\tilde{y}_j$. Define $\tilde{u}_j$ as $\tilde{y}_j$ if it is a non-leech, and as the host of $\tilde{y}_j$ otherwise. By the induction hypothesis, $\tilde{u}_j$ is large, and so there are at least $k+1$ clean 10-friends in $F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)$. Moreover, all these points will also belong to $F(\tilde{u}_j)$ by Claim 3.13, unless $|F(\tilde{u}_j)| = 3k+3$. Recall that x'_j and y'_j are 24-friends. Hence

$$\delta(\tilde{x}_j, \tilde{y}_j) \;\leq\; \delta(\tilde{x}_j, x'_j) + \delta(x'_j, y'_j) + \delta(y'_j, \tilde{y}_j) \;\leq\; 3 \cdot 5^{\tilde{l}_j+1} + 24 \cdot 5^{\tilde{l}_j+1} + 3 \cdot 5^{\tilde{l}_j+1} \;=\; 30 \cdot 5^{\tilde{l}_j+1} \;=\; 150 \cdot 5^{\tilde{l}_j}.$$

Note also that $\delta(\tilde{y}_j, \tilde{u}_j) \leq 24 \cdot 5^{\tilde{l}_j}$, and so $\tilde{x}_j$ and $\tilde{u}_j$ are 174-friends. By construction, there is a cross edge between $\tilde{x}_j$ and $\tilde{u}_j$ (assuming $\gamma \geq 174$), and so all points of $F(\tilde{u}_j)$ will belong to $F(x)$ by the construction. Since $|F(x)| < 3k+3$, it must hold that $|F(\tilde{u}_j)| < 3k+3$. It follows that all the at least $k+1$ points of $F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)$ belong to $F(\tilde{u}_j)$ (and thus also to $F(x)$). On the other hand, recall that $S(\tilde{y}_j) \cap S(x) = \emptyset$.

Since $S(\tilde{u}_j) = S(\tilde{y}_j)$ and $S(\tilde{x}_j) = S(x)$, we have $\tilde{u}_j \neq \tilde{x}_j$. Claim 3.10 implies that $F(a(x)) \cap F(a(\tilde{u}_j)) = \emptyset$, and so all the at least $k+1$ points in $F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)$ belong to $F(\tilde{u}_j)$ but do not belong to $F(a(x))$.

Suppose first that $\tilde{l}_j$ is not the last level of $term(\tilde{u}_j)$. In this case all the at least $k+1$ points in $F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)$ remain clean at level $\tilde{l}_j + 1$ by Claim 3.13. However, it is not difficult to see that all these points are 10-friends of the $(\tilde{l}_j + 2)$-level copy of x. Hence, by Claim 3.12, these points will remain clean during the entire $term(x)$. As mentioned, all points of $F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)$ belong to both $F(\tilde{u}_j)$ and $F(x)$, but not to $F(a(x))$. Consequently, at least $k+1$ points from $F(x)$ do not belong to $F(a(x))$, and we have $|F(x)| \geq |F(a(x)) \setminus S(a(x))| + |F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)| \geq 2k+2$. This completes the proof in this case.

In what follows we assume that $\tilde{l}_j$ is the last level of $term(\tilde{u}_j)$. Note that $F(\tilde{u}_j) = F(\tilde{u}_j) \setminus S(\tilde{u}_j)$. By the induction hypothesis, $|F(\tilde{u}_j)| \geq 2k+2$. By Claim 3.12, all points of $F(\tilde{u}_j)$ remain clean at level $\tilde{l}_j$. The analysis splits into two subcases.
Case 2.a: $F(a(x)) \cap F(\tilde{u}_j) = \emptyset$. In this case some of the points in $F(\tilde{u}_j)$ may get dirty at level $\tilde{l}_j + 1$, which occurs if they are appointed as new surrogates of some $(\tilde{l}_j + 1)$-level node. We argue that no more than $k+1$ of these points may get dirty at level $\tilde{l}_j + 1$. Suppose for contradiction otherwise. Since any appointing node appoints exactly $k+1$ points as new surrogates, this means that at least two distinct $(\tilde{l}_j + 1)$-level appointing nodes v_1 and v_2 must appoint points from $F(\tilde{u}_j)$ as their new surrogates. Since all points of $F(\tilde{u}_j)$ are within distance $10 \cdot 5^{\tilde{l}_j} = 2 \cdot 5^{\tilde{l}_j + 1}$ from $\tilde{u}_j$, it follows that they are all within distance $4 \cdot 5^{\tilde{l}_j+1}$ from each other. Since v_1 (respectively, v_2) appoints at least one point p_1 (resp., p_2) from $F(\tilde{u}_j)$ as a new surrogate, we have $\delta(v_1, p_1), \delta(v_2, p_2) \leq 10 \cdot 5^{\tilde{l}_j+1}$. Hence

$$\delta(v_1, v_2) \;\leq\; \delta(v_1, p_1) + \delta(p_1, p_2) + \delta(p_2, v_2) \;\leq\; 10 \cdot 5^{\tilde{l}_j+1} + 4 \cdot 5^{\tilde{l}_j+1} + 10 \cdot 5^{\tilde{l}_j+1} \;=\; 24 \cdot 5^{\tilde{l}_j+1}.$$

Hence v_1 and v_2 are 24-friends, and so one would have to become a leech of the other and would not be an appointing node, a contradiction. Since $|F(\tilde{u}_j)| \geq 2k+2$, it follows that at least $k+1$ points from $F(\tilde{u}_j)$ must be clean at the beginning of level $\tilde{l}_j + 2$. Moreover, it is not difficult to see that all these points are 10-friends of the $(\tilde{l}_j + 2)$-level copy of x. Hence, they will remain clean during the entire $term(x)$ by Claim 3.12, and will belong to $F(x)$. We have proved that $F(x)$ contains at least $2k+2$ points, at least $k+1$ of which do not belong to $F(a(x))$, which completes the proof in this case.
Case 2.b: $F(a(x)) \cap F(\tilde{u}_j) \neq \emptyset$. In this case $\tilde{u}_j$ must be a 20-friend of the $\tilde{l}_j$-level copy $\tilde{x}_j$ of x. It is easy to see that all points of $F(\tilde{u}_j)$ will be 10-friends of the $(\tilde{l}_j + 1)$-level copy of x, and so they will remain clean during the entire $term(x)$ by Claim 3.12. By construction, $F(x)$ will contain all points of $F(\tilde{u}_j)$. We have also shown that all the at least $k+1$ points of $F(a(\tilde{u}_j)) \setminus S(\tilde{u}_j)$ belong to $F(\tilde{u}_j)$. Finally, Claim 3.10 implies that $F(a(x)) \cap F(a(\tilde{u}_j)) = \emptyset$, and we are done. Lemma 3.14 follows. ▮

Lemma 3.14 yields the following corollary, which, in turn, implies that our construction is well-defined.

Corollary 3.15 (Procedure $ComputeSets_{(i)}$ **is Well-Defined)** *For any appointing node x, we have $|F(x)| \geq 2k+2$.*

Proof: By construction, the statement clearly holds for both leaf nodes and nodes for which all children are clean. We henceforth assume that x is an i-level appointing node, for $i \geq 1$, with at least one dirty child y. Define z as y if it is a non-leech, or as y's host otherwise. We argue that $term(z)$ must be over at level $i-1$. Indeed, otherwise z's parent will be a dirty non-leech, and so x will not become an appointing node, a contradiction. (This follows immediately from the construction if y is a non-leech. If y is a leech and z is its host, then x would be a 24-friend of z's parent by Claim 3.2, and would become a leech of either z's parent or some other i-level dirty non-leech 24-friend of x, and not an appointing node.)

We have shown that $term(z)$ is over at level $i-1$. Hence, Lemma 3.14 implies that $|F(z)| \geq 2k+2$. Moreover, all points of $F(z)$ must remain clean at level $i-1$ by Claim 3.12. It is easy to see that all points of $F(z)$ are 10-friends of x. Consequently, all these points must belong to $F(x)$ by the construction, unless $|F(x)| = 3k+3$. The corollary follows. ▮

3.2.3 Completing the Degree Analysis. In this section we prove that the degree of our FT spanner $\mathcal{H}$ is at most $\epsilon^{-O(d)} \cdot k$. We do this in two stages. In the first stage we bound the degree of $\mathcal{H}$ due to cross edges, and in the second stage we show that the degree contribution due to tree edges is negligible.

First Stage. Denote the degree of an arbitrary point p due to cross edges until level i by $\mathsf{deg}_i(p)$. Recall that $\xi = \epsilon^{-O(d)}$ is an upper bound for the maximum degree of any tree node in the basic spanner H. Since the surrogates of any node are 34-friends of that node (see Observation 3.9) and the set of all i-level nodes constitutes a 5^i-packing, for each $i \in [0, \ell]$, Fact 2.1 implies that any point can serve as a surrogate of at most $O(1)^{O(d)}$ nodes at each level. By construction, the number of surrogates in any node is at most $2k+1$, and so a bipartite clique that replaces a cross edge may increase the degree of each of the relevant surrogates by at most $2k+1$ units. It follows that the degree of any surrogate (due to cross edges) may increase by at most $\xi \cdot O(1)^{O(d)} \cdot (2k+1) \leq \xi^2(2k+1)$ units at each level.

Recall that $D = (\tau+4)\xi^2(2k+1)$, and that ℓ is the last level in the net-tree T (the level of the root).

In what follows we show that $\mathsf{deg}_\ell(p) \leq 2D$, for any point $p \in X$.

Lemma 3.16 *For any surrogate $p \in S(x)$ of a j-level clean node x, we have $\mathsf{deg}_j(p) \leq D$.*

Proof: Let p be the point in $S(x)$ of maximum degree, i.e., for every point $q \in S(x)$, $\mathsf{deg}_j(q) \leq \mathsf{deg}_j(p)$. By Observation 3.5, $p \in D(x)$, implying that x is the j-level base bag of p.
If $\mathsf{deg}_j(p) < 2k+2$, then we are done. We henceforth assume that $\mathsf{deg}_j(p) \geq 2k+2$. Let i be the minimum level such that $\mathsf{deg}_i(p) \geq 2k+2$, and let $q_1, \ldots, q_{2k+2}$ be $2k+2$ arbitrary neighbors of p due to cross edges until level i; observe that $2k+2 \leq \mathsf{deg}_i(p) \leq 2k+1+\xi^2(2k+1)$. By Claim 3.3, p is within distance $(\gamma+68)5^i$ from each of the $2k+2$ points $q_1, \ldots, q_{2k+2}$. Moreover, Claim 3.3 implies that all $2k+2$ points $q_1, \ldots, q_{2k+2}$ will be 10-friends of the $(i+\tau)$-level base bag y of p. Write $i' = i+\tau$.
If all these $2k+2$ points are clean at the beginning of level i', then they would all belong to $F(y)$ by the construction, unless $|F(y)| = 3k+3$. (Indeed, Corollary 3.7 implies that any node y_l for which $q_l \in S(y_l)$ at level until i' must be clean, for each $l \in [2k+1]$, and so we have $q_l \in F(y_l)$ by Observation 3.8. Since there is a cross edge between p and q_l in our FT spanner $\mathcal{H}$ at level at most $i \leq i'$, it follows that q_l will be added to the reserve set of the corresponding descendant of y, and will eventually be added to $F(y)$, unless $|F(y)| = 3k+3$.) In either case $|F(y)| \geq 2k+2$, and so y will be dirty (in fact, it will be large). By Observation 3.4, $\pi(y)$ and $\pi(\pi(y))$ will be dirty as well.
Otherwise, at least one point q_l becomes dirty before level i'. Let w be the (dirty) appointing node which made q_l dirty, where $q_l \in S(w)$, and let w' be the i'-level ancestor of w. By Observation 3.4, w' must be dirty. Note that q_l is a 10-friend of w, and so it is also a 10-friend of w' by Observation 3.1. Since q_l is a 10-friend of both y and a w', it follows that y and w' are 20-friends. If w' is a non-leech, then y will become its leech and thus dirty. By Observation 3.4, $\pi(y)$ and $\pi(\pi(y))$ will be dirty as well. Otherwise w' is a leech of some host z, and so w' and z are 24-friends. There are two cases.
Case 1: term(z) is not over at level i'. By construction, $\pi(z)$ will be a dirty non-leech. Observe that

$$\delta(\pi(y),\pi(z)) \;\leq\; \delta(\pi(y),y)+\delta(y,w')+\delta(w',z)+\delta(z,\pi(z)) \;\leq\; 3\cdot 5^{i'+1}+20\cdot 5^{i'}+24\cdot 5^{i'}+3\cdot 5^{i'+1} \;<\; 15\cdot 5^{i'+1}.$$

Hence $\pi(y)$ is a 15-friend of $\pi(z)$. In other words, $\pi(y)$ is close enough to become a leech of $\pi(z)$, and so it must be dirty. By Observation 3.4, $\pi(\pi(y))$ will be dirty too.
Case 2: term(z) is over at level i'. By Lemma 3.14, we have $|F(z)| \geq 2k+2$. Also, all the points in $F(z) = F(z) \setminus S(z)$ are clean at the beginning of level i', and will remain clean at level i' by Claim 3.12. For any such point s, we have

$$\delta(\pi(\pi(y)),s) \;\leq\; \delta(\pi(\pi(y)),y)+\delta(y,z)+\delta(z,s) \;\leq\; 4\cdot 5^{i'+2}+44\cdot 5^{i'}+10\cdot 5^{i'} \;<\; 7\cdot 5^{i'+2}.$$

In other words, all points in $F(z)$ are 7-friends of $\pi(\pi(y))$.
If all these points are clean at the beginning of level $i'+2$, then they will belong to $F(\pi(\pi(y)))$ by the construction, unless $|F(\pi(\pi(y)))| = 3k+3$. (Indeed, there is a cross edge between y and z (assuming $\gamma \geq 44$), and so all points of $F(z)$ will be added to the reserve set of y, and will eventually be added

to $F(\pi(\pi(y)))$, unless $|F(\pi(\pi(y)))| = 3k+3$.) Hence $\pi(\pi(y))$ will either become a leech of some dirty $(i'+2)$-level node or it will become an appointing node. In either case $\pi(\pi(y))$ will be dirty.
Otherwise, this means that some $(i'+1)$-level appointing node v made at least one of these points $s \in F(z)$ dirty at level $i'+1$. Since v is an appointing node, both v and $\pi(v)$ must be dirty non-leeches by the construction. The fact that v appoints s as a new surrogate implies that v and s are 10-friends. Hence

$$\delta(\pi(v), s) \ \leq \ \delta(\pi(v), v) + \delta(v, s) \leq 3 \cdot 5^{i'+2} + 10 \cdot 5^{i'+1} = 5 \cdot 5^{i'+2}.$$

It follows that $\pi(v)$ is a dirty non-leech $(i'+2)$-level node, which is a 5-friend of s. Since s is a 7-friend of $\pi(\pi(y))$, we conclude that $\pi(\pi(y))$ is a 12-friend of $\pi(v)$. In other words, $\pi(\pi(y))$ is close enough to become a leech of $\pi(v)$, and so it must be dirty.

Summarizing, we have shown that $\pi(\pi(y))$ is dirty. Note that $\pi(\pi(y))$ is the $(i'+2)$-level base bag of p and recall that x is the j-level base bag of p. Since x is clean, Observation 3.4 implies that $j \leq i'+1 = i+\tau+1$. Recall that $\mathsf{deg}_i(p) \leq 2k+1+\xi^2(2k+1)$. Also, in each of the $j-i \leq \tau+1$ levels $i+1, \ldots, j$, the degree of p may increase by at most $\xi^2(2k+1)$ units, and so

$$\mathsf{deg}_j(p) \ \leq \ \mathsf{deg}_i(p) + (\tau+1)\xi^2(2k+1) \ \leq \ 2k+1+\xi^2(2k+1) + (\tau+1)\xi^2(2k+1) \ \leq \ D. \qquad \blacksquare$$

Let i be the first level in which p becomes dirty. (If no such i exists, we can define $i = \ell+1$.)

By construction, p is appointed as a new surrogate of a large (and thus dirty) i-level appointing node x. Corollary 3.7 implies that p cannot be a surrogate of dirty nodes before level i, and so $\mathsf{deg}_{i-1}(p) \leq D$ must hold by Lemma 3.16. In the degenerate case when p is not appointed as a surrogate of dirty nodes (i.e., $i = \ell+1$), we have $\mathsf{deg}_\ell(p) = \mathsf{deg}_{i-1}(p) \leq D$, and we are done. We henceforth assume that $i \leq \ell$.

Recall that p is appointed at level i for a term of at least $\tau+3$ levels, which consists of two phases. The first phase starts at level i, and ends when the degree of p due to (non-redundant) cross edges since its appointment at level i reaches $k+1$. When the second phase starts, the degree of p is between $k+1$ and $k+\xi^2(2k+1)$. The second phase lasts $\tau+2$ levels. In each of these levels p may incur $\xi^2(2k+1)$ additional units to its degree. Consequently, when the term of p is over, the degree of p due to cross edges since its appointment is bounded above by $k+\xi^2(2k+1)+(\tau+2)\xi^2(2k+1) \leq D$. (In fact, it is possible that p does not finish its term due to abundance of surrogates; see the second remark at the end of Section 3.1.4. Thus we can refer to $term(x)$ instead of the term of p.) When p was appointed as a new surrogate of the large i-level node x (at level i), it was marked as dirty. Therefore, by construction, p will not be re-appointed again as a surrogate (of either clean nodes or dirty ones) after $term(x)$ is over.

Summarizing, the degree of p may increase by at most D units while it is a surrogate of clean nodes, and by at most D additional units while it is a surrogate of dirty nodes. It follows that $\mathsf{deg}_\ell(p) \leq 2D$.

Second Stage. Consider a tree edge $(x, \pi(x))$ between a node $x = (p, j-1)$ and its parent $\pi(x) = (q, j)$ in the tree. This edge is translated into a bipartite clique C_x between $S(x)$ and $S(\pi(x))$ in our FT spanner $\mathcal{H}$. If $S(x) = S(\pi(x))$, then this edge is *redundant* and we do not replace it by a bipartite clique.

Next, we argue that the degree of a point p in $\mathcal{H}$ due to tree edges is small. To this end we analyze the degree of p over all cliques C_x that correspond to tree edges $(x, \pi(x))$. We say that C_x is a *j-level clique* if $\pi(x)$ is a j-level node (that is, the level of the clique C_x is determined by the level of $\pi(x)$).

Let i be the first level in which p becomes dirty. If p does not become dirty, then we define $i = \ell+1$. By Corollary 3.7, p may only be a surrogate of clean nodes in levels $0, \ldots, i-1$, whereas it may only be a surrogate of dirty nodes in levels $i, \ldots, \ell$. Note that the j-level base bag of p, for $j < i$, is not necessarily clean. Let l be the first level in which the l-level base bag of p is dirty, and assuming $l \geq 1$, let z denote the $(l-1)$-level base bag of p. Notice that all ancestors of z are base bags of p, and they are dirty by Observation 3.4. Since z is clean, Observation 3.4 implies that all the descendants of z are clean too. Claim 3.6 implies that $l \leq i$. Moreover, p cannot be a surrogate of any j-level node, for all $j \in [l, i-1]$.

We first analyze the degree of p over all cliques until level $l-1$, i.e., over j-level cliques C_x that correspond to tree edges $(x, \pi(x))$, where $\pi(x)$ is a j-level node and $j \leq l-1$, and p belongs to either $S(x)$

or $S(\pi(x))$. Observation 3.5 implies that $\pi(x)$ is the j-level base bag of p. Consequently, both x and $\pi(x)$ must be clean. Notice that $\pi(x)$ must be a descendant of the $(l-1)$-level base bag z of p. Observation 3.5 implies that $S(x) \subseteq S(\pi(x)) \subseteq S(z)$. Consequently, the clique C_x that corresponds to the edge $(x, \pi(x))$ is contained in the *internal clique* over $S(z)$, which includes an edge between any pair of points from $S(z)$. It follows that all cliques C_x that may increase p's degree until level $l-1$ are contained in the internal clique over $S(z)$. Corollary 3.8 implies that $|S(z)| < 2k+2$. Hence the degree of p over all cliques C_x until level $l-1$ is bounded above by $2k$.

For a fixed j, the degree of p due to all j-level cliques C_x may increase by at most $O(1)^{O(d)} \cdot (2k+1)$ units. (This is because (i) p may serve as a surrogate of at most $O(1)^{O(d)}$ nodes at each level, (ii) each tree node has at most $O(1)^{O(d)}$ children, and (iii) each node has at most $2k+1$ surrogates.) In particular, the degree of p due to all l-level cliques (where x and $\pi(x)$ are in levels $l-1$ and l, respectively) is at most $O(1)^{O(d)} \cdot (2k+1)$. Similarly, the degree of p due to all i-level cliques (where x and $\pi(x)$ are in levels $i-1$ and i, respectively) is also in check. Recall also that p cannot be a surrogate of any j-level node, for all $j \in [l, i-1]$. Consequently, the degree of p over all cliques C_x until level i is at most $2k + O(1)^{O(d)} \cdot (2k+1)$.

In what follows we analyze the degree of p over all cliques of level at least $i+1$, i.e., over j-level cliques C_x that correspond to tree edges $(x, \pi(x))$, where $\pi(x)$ is a j-level node, for $j \geq i+1$, and p belongs to either $S(x)$ or $S(\pi(x))$.

By definition, p is appointed as a surrogate of a large i-level node, denoted y, and subsequently becomes dirty. The degree of p may increase due to tree edges during *term*(y). However, our construction guarantees that p will not become a surrogate of any node after *term*(y) is over. Let i' denote the last level of *term*(y), let y' be the i'-level ancestor of y, and let $\Pi = \Pi_{y,y'}$ be the path between y and y' in the net-tree T. Observe that cliques of level larger than $i'+1$ cannot increase the degree of p, and so it suffices to bound the degree of p over all cliques of level between $i+1$ and $i'+1$.

In level $i'+1$ (which is the last level of interest to us) the degree of p will increase by at most $O(1)^{O(d)} \cdot (2k+1)$ units. Consider now a j-level clique C_x that corresponds to a tree edge $(x, \pi(x))$, where $\pi(x)$ is a j-level node, for $j \in [i+1, i']$, and p belongs to either $S(x)$ or $S(\pi(x))$. We also assume that the edge $(x, \pi(x))$ is non-redundant, i.e., $S(x) \neq S(\pi(x))$. We argue that the clique C_x is *incarnated* in some clique that corresponds to either a $(j-1)$-level cross edge or a j-level cross edge. Since we have already shown that the degree of p due to cross edges is in check, proving this assertion would imply that the degree of p due to tree edges is in check as well, thus completing the degree analysis.

Observe that all nodes along the path Π are dirty non-leech copies of y. Moreover, by Claim 3.11, for any j-level node v, where $j \in [i, i']$, either $S(v) = S(y)$ or $S(v) \cap S(y) = \emptyset$ must hold, and in the former case Claim 3.10 implies that v is either the j-level copy of y or one of its leeches. In particular, if p belongs to $S(v)$, then we have $S(v) = S(y)$. Consequently, if p belongs to both $S(x)$ and $S(\pi(x))$, then we have $S(x) = S(\pi(x)) = S(y)$, and so the edge $(x, \pi(x))$ is redundant. We henceforth assume that p belongs to either $S(x)$ or $S(\pi(x))$, but not to both, and so $S(x) \cap S(\pi(x)) = \emptyset$. This means that x cannot belong to Π, as otherwise $\pi(x)$ would belong to Π too, and the edge $(x, \pi(x))$ would be redundant.

Suppose next that x does not belong to Π but $\pi(x)$ belongs to Π. In this case $\pi(x)$ has a dirty non-leech child u in Π, with $S(u) = S(\pi(x)) = S(y)$. Since x and u are siblings, they must be γ-friends (assuming $\gamma \geq 30$). Hence they are connected by a cross edge in the basic spanner H, and so C_x is incarnated in the clique $C_{x,u}$ that corresponds to the $(j-1)$-level cross edge (x, u).

We may henceforth assume that neither x nor $\pi(x)$ belong to Π. The first case is that p belongs to $S(x)$ but does not belong to $S(\pi(x))$. In this case x is a leech of the $(j-1)$-level copy y_{j-1} of y, and so $\delta(x, y_{j-1}) \leq 24 \cdot 5^{j-1}$. Let y_j denote the j-level copy of y, and note that $y_j = \pi(y_{j-1})$. We have

$$\delta(y_j, \pi(x)) \leq \delta(y_j, y_{j-1}) + \delta(y_{j-1}, x) + \delta(x, \pi(x)) \leq 3 \cdot 5^j + 24 \cdot 5^{j-1} + 3 \cdot 5^j < 11 \cdot 5^j.$$

Hence $\pi(x)$ is an 11-friend of the dirty non-leech y_j, and so it must be a leech too by the construction. Note that the host h of $\pi(x)$ cannot be y_j, otherwise the edge $(x, \pi(x))$ would be redundant. Observe also that $\delta(y_j, h) \leq \delta(y_j, \pi(x)) + \delta(\pi(x), h) \leq 11 \cdot 5^j + 24 \cdot 5^j = 35 \cdot 5^j$. In other words, y_j and h are

γ-friends (assuming $\gamma \geq 35$). Thus they are connected by a cross edge in the basic spanner H, and so C_x is incarnated in the clique $C_{y_j,h}$ that corresponds to the j-level cross edge (y_j, h).

The remaining case is that p does not belong to $S(x)$ but belongs to $S(\pi(x))$. Thus $\pi(x)$ is a leech of the j-level copy y_j of y, and so $\delta(y_j, \pi(x)) \leq 24 \cdot 5^j$ and $S(\pi(x)) = S(y_j) = S(y_{j-1})$. We have

$$\delta(y_{j-1}, x) \;\leq\; \delta(y_{j-1}, y_j) + \delta(y_j, \pi(x)) + \delta(\pi(x), x) \;\leq\; 3 \cdot 5^j + 24 \cdot 5^j + 3 \cdot 5^j \;=\; 150 \cdot 5^{j-1}.$$

In other words, y_{j-1} and x are γ-friends (assuming $\gamma \geq 150$). Thus they are connected by a cross edge in the basic spanner H, and so C_x is incarnated in the clique $C_{y_{j-1},x}$ that corresponds to the $(j-1)$-level cross edge (y_{j-1}, x). This concludes the proof of the above assertion and completes the degree analysis.

3.3 Fault-Tolerance and Stretch

In this section we show that $\mathcal{H}$ is a k-FT $(1+\epsilon)$-spanner for (X, δ). To this end we show that for any pair $p, q \in X \setminus F$ of functioning points (where $|F| \leq k$), there is a $(1+\epsilon)$-spanner path in $\mathcal{H} \setminus F$.

Consider the $(1+\epsilon)$-spanner path $\Pi_{p,q}$ between p and q in the basic spanner H that is guaranteed by Lemma 2.2. The path $\Pi_{p,q}$ is obtained by climbing up from the leaf nodes $(p, 0)$ and $(q, 0)$ to some j-level ancestors (p', j) and (q', j), respectively, which are connected by a cross edge $((p', j), (q', j))$. By Observation 3.5, p is a surrogate of every clean node along the sub-path of $\Pi_{p,q}$ that climbs from $(p, 0)$ to (p', j). Also, every dirty node along this path is complete, hence it has $k+1$ surrogates, at least one of which must function. Since every edge along this path is translated into a bipartite clique in $\mathcal{H}$ between the corresponding surrogate sets, we obtain this way a functioning path that starts at p and ends at an arbitrarily chosen surrogate $\tilde{p} \in S(p', j) \setminus F$ of (p', j). Similarly, we obtain a functioning path that starts at q and ends at an arbitrarily chosen surrogate $\tilde{q} \in S(q', j) \setminus F$ of (q', j). The cross edge between $\tilde{p}$ and $\tilde{q}$ in $\mathcal{H}$ is functioning too, i.e., $(\tilde{p}, \tilde{q}) \in \mathcal{H} \setminus F$. (In the degenerate case where $S(p', j) = S(q', j)$, the cross edge between (p', j) and (q', j) in H is redundant, and we do not translate it into a bipartite clique in $\mathcal{H}$. Hence, the edge $(\tilde{p}, \tilde{q})$ may not belong to $\mathcal{H}$. However, in this case we can simply take $\tilde{p} = \tilde{q}$.) In this way we obtain a functioning path $s(\Pi_{p,q})$ between p and q in our FT spanner $\mathcal{H}$. Observation 3.9 implies that any node along the original path $\Pi_{p,q}$ in H is a 34-friend of all its surrogates (and in particular, of all its functioning surrogates). Lemma 2.3 implies that $s(\Pi_{p,q})$ is a $(1+O(\epsilon))$-spanner path between p and q, but we can reduce the stretch to $1+\epsilon$ by scaling γ up by some constant.

4 Lightness

The radius $rad(x)$ of an i-level node is its distance scale; we have $rad(x) = 5^i$, but in most other papers (such as [26, 12]) $rad(x)$ is equal to 2^i. The standard analysis of [12] (see also [14, 44, 15]) implies that the sum of radii of all tree nodes is $O(\log n) \cdot \omega(MST(X))$. Since the degree of any tree node in the basic spanner construction H is $\epsilon^{-O(d)}$ and since all edges in H that are incident on a node x have weight at most $O(\frac{1}{\epsilon}) \cdot rad(x)$, it follows that the lightness of H is bounded above by $\epsilon^{-O(d)} \cdot \log n$. Our FT spanner construction $\mathcal{H}$ replaces each edge (x, y) in H by a bipartite clique of size $O(k^2)$ between $S(x)$ and $S(y)$. By Observation 3.9, the weight of each of the $O(k^2)$ edges of the clique exceeds the weight $\omega(x, y)$ of the original edge by an additive factor of at most $34 \cdot (rad(x) + rad(y)) = O(rad(x) + rad(y))$. Hence the lightness of our FT spanner $\mathcal{H}$ is bounded by $\epsilon^{-O(d)} \cdot k^2 \log n$. (See [15] for a detailed argument.)

To obtain the desired lightness bound of $\epsilon^{-O(d)}(k^2 + k \log n)$, we need to work harder. The key idea is to replace each bipartite clique by a bipartite matching. We can do this "matching replacement" only for edges (x, y) where both nodes x and y are complete. Note that complete nodes are not necessarily dirty. For technical reasons it is more convenient to do this matching replacement for edges (x, y) where both nodes x and y are dirty. In particular, the surrogate set of a dirty node contains exactly $k+1$ points (see Observation 3.4). In this case $|S(x)| = |S(y)| = k+1$, and we will use a perfect bipartite matching (of size $k+1$) to connect $S(x)$ with $S(y)$. If x or y (or both) are clean, we will connect $S(x)$ and $S(y)$ by a bipartite clique as before. Using a bipartite matching instead of a bipartite clique shaves a factor of k in the lightness bound; more specifically, the total lightness of all the bipartite matchings is $\epsilon^{-O(d)} \cdot k \log n$.

Next, we analyze the lightness of the bipartite cliques. Each such clique corresponds to an edge (x, y) of H, in which either x or y (or both) are clean. We charge the total weight of all the clique edges with the clean nodes among x and y. In what follows we bound the total charge W over all clean nodes.

Recall that all descendants of a clean node are clean as well (see Observation 3.4). For a clean node x, let $W(x)$ denote the total charge over x and all its descendants. In other words, $W(x)$ stands for the total weight of clique edges that correspond to edges of H that are incident on either x or its descendants.

Lemma 4.1 *For a clean i-level node x, we have $W(x) \le \epsilon^{-O(d)} \cdot k^2 \cdot rad(x)$.*

Proof: Let $m(x)$ denote the number of edges in $\mathcal{H}$ that correspond to edges of H that are incident on either x or its descendants. Observation 3.5 implies that for any descendant x' of x, $S(x') \subseteq S(x)$. It follows that $m(x) \le \sum_{p \in S(x)} \widehat{\mathsf{deg}}_i(p)$, where $\widehat{\mathsf{deg}}_i(p)$ stands for the total degree of a point p due to cross edges until level i and due to tree edges until level $i+1$ (i.e., including the edge $(x, \pi(x))$). Recall that $\mathsf{deg}_i(p)$ stands for the degree of a point p due to cross edges until level i. By Lemma 3.16, $\mathsf{deg}_i(p) \le D$, for each $p \in S(x)$. We have also shown that the degree of such a point p due to tree edges until level $i+1$ (including the edge $(x, \pi(x))$) is bounded above by $2k + O(1)^{O(d)} \cdot (2k+1)$. We thus have $\widehat{\mathsf{deg}}_i(p) \le \mathsf{deg}_i(p) + 2k + O(1)^{O(d)} \cdot (2k+1) \le \epsilon^{-O(d)} \cdot k$. By construction, $|S(x)| \le 2k+1$. It follows that

$$m(x) \le \sum_{p \in S(x)} \widehat{\mathsf{deg}}_i(p) \le \sum_{p \in S(x)} \epsilon^{-O(d)} \cdot k \le (2k+1) \cdot (\epsilon^{-O(d)} \cdot k) \le \epsilon^{-O(d)} \cdot k^2.$$

By Claim 3.3, each of the cross edges in $\mathcal{H}$ that increase p's degree until level i has weight at most $(\gamma + 68)5^i$. (In fact, since x is clean, $S(x) = D(x)$. Hence the distance between x and its surrogates is at most $4 \cdot 5^i$, and so the weight of these edges is at most $(\gamma + 4 + 34)5^i = (\gamma + 38)5^i$.) Also, since the distance between any two surrogates of $S(x)$ is at most $8 \cdot 5^i$, each of the tree edges in $\mathcal{H}$ that increase p's degree until level i has weight at most $8 \cdot 5^i$. By Observation 3.9, the surrogates of $\pi(x)$ are 34-friends of it, hence the edges of $\mathcal{H}$ that correspond to the tree edge $(x, \pi(x))$ in H have weight at most $4 \cdot 5^i + 3 \cdot 5^{i+1} + 34 \cdot 5^{i+1} = 189 \cdot 5^i \le (\gamma + 68)5^i$; this inequality holds for $\gamma \ge 121$. Recall also that $rad(x) = 5^i$. We conclude that the total weight of all $m(x)$ edges is at most $m(x)(\gamma + 68)5^i = \epsilon^{-O(d)} \cdot k^2 \cdot rad(x)$. ▮

A clean node is called *almost-dirty* if its parent in T is dirty. Let A be the set of all almost-dirty nodes in T. By Observation 3.4, for any pair $x, y \in A$ of distinct almost-dirty nodes, neither one of them can be an ancestor of the other. Moreover, any clean node $z \notin A$ that is not almost-dirty is a descendant of a single almost-dirty node, which implies that the total charge W of all clean nodes is equal to $\sum_{x \in A} W(x)$. By Lemma 4.1, we have $W = \sum_{x \in A} W(x) \le \epsilon^{-O(d)} \cdot k^2 \cdot \sum_{x \in A} rad(x)$.

Next, we show that $\sum_{x \in A} rad(x) = O(\omega(MST(X)))$, which implies that $W \le \epsilon^{-O(d)} \cdot k^2 \cdot \omega(MST(X))$. Since the total weight of the bipartite cliques is at most W, the desired lightness bound would follow.

The following lemma implies that the net-points $p(x), p(y)$ that correspond to a pair $x, y \in A$ of distinct almost-dirty nodes are far away from each other, and in particular distinct (i.e., $p(x) \ne p(y)$).

Lemma 4.2 (Almost-Dirty Nodes are Far Away From Each Other) *For any pair $(p, i), (q, j)$ of distinct nodes from A, with $i \le j$, we have $\delta(p, q) \ge 5^{j-1} = \frac{rad(q,j)}{5}$.*

Proof: Notice that both p and q belong to the i-level net N_i. Since N_i a 5^i-packing, we have $\delta(p, q) \ge 5^i$. If $i \ge j - 1$, then we are done. We henceforth assume that $i \le j - 2$.

Suppose for contradiction that $\delta(p, q) < 5^{j-1}$. Let $(r, j-1)$ be the $(j-1)$-level ancestor of (p, i). Since $j - 1 \ge i + 1$, Observation 3.4 implies that $(r, j-1)$ must be dirty. We have $\delta(r, p) \le 4 \cdot 5^{j-1}$, and so $\delta(r, q) \le \delta(r, p) + \delta(p, q) \le 4 \cdot 5^{j-1} + 5^{j-1} = 5 \cdot 5^{j-1}$. Note also that $(q, j-1)$ is a clean child of (q, j).

If $(r, j-1)$ is a non-leech, then $(q, j-1)$ would have to become its leech (since $(q, j-1)$ and $(r, j-1)$ are 5-friends) and thus dirty, a contradiction. We may henceforth assume that $(r, j-1)$ is a leech of some

$(j-1)$-level node $(h, j-1)$. By construction, we have $\delta(r,h) \leq 24 \cdot 5^{j-1}$. There are two cases.
Case 1: The parent $(\pi(h), j)$ of $(h, j-1)$ is a non-leech. Notice that

$$\delta(\pi(h), q) \ \leq \ \delta(\pi(h), h) + \delta(h, r) + \delta(r, q) \ \leq \ 3 \cdot 5^j + 24 \cdot 5^{j-1} + 5 \cdot 5^{j-1} \ < \ 9 \cdot 5^j.$$

This means that (q, j) is a 9-friend of $(\pi(h), j)$, and would become its leech and thus dirty, a contradiction.
Case 2: The parent $(\pi(h), j)$ of $(h, j-1)$ is a leech. By construction, the term of the surrogates of $(h, j-1)$ must be over at level $j-1$. Lemma 3.14 implies that $|F(h, j-1)| \geq 2k+2$. Also, all points of $F(h, j-1)$ remain clean at level $j-1$ by Claim 3.12. By construction, each point of $F(h, j-1)$ is a (clean) 10-friend of $(h, j-1)$. Moreover, each point $s \in F(h, j-1)$ is an 8-friend of (q, j); to see this, note that

$$\delta(s, q) \ \leq \ \delta(s, h) + \delta(h, r) + \delta(r, q) \ \leq \ 10 \cdot 5^{j-1} + 24 \cdot 5^{j-1} + 5 \cdot 5^{j-1} \ < \ 8 \cdot 5^j.$$

Observe also that $\delta(h, q) \leq \delta(h, r) + \delta(r, q) \leq 24 \cdot 5^{j-1} + 5 \cdot 5^{j-1} = 29 \cdot 5^{j-1}$. Hence $(h, j-1)$ and $(q, j-1)$ are 29-friends, and so there is a cross edge between them in the basic spanner H (assuming $\gamma \geq 29$). By construction, all the at least $2k+2$ points of $F(h, j-1)$ will be added to the reserve set $R(q, j-1)$ of $(q, j-1)$. Moreover, all these points will be subsequently added to $F(q, j)$, unless $|F(q, j)| = 3k+3$. In any case (q, j) will be large, and thus dirty, a contradiction. The lemma follows. ∎

Let $X_A \subseteq X$ be the set of net-points that correspond to nodes in A, i.e., $X_A = \{p(x) \mid x \in A\}$. Consider the $MST(X_A)$ of (X_A, δ), and note that $\omega(MST(X_A))) \leq 2 \cdot \omega(MST(X))$. Let rt be an arbitrary node in A. We root $MST(X_A)$ at $p(rt)$, and orient all the edges towards $p(rt)$. In this way for each node $x \in A \setminus \{rt\}$, the corresponding net-point $p(x)$ has a single outgoing edge in $MST(X_A)$, denoted $e(x)$. By Lemma 4.2, $\omega(e(x)) \geq \frac{rad(x)}{5}$. Also, for any pair $x, y \in A$ of distinct almost-dirty nodes, their corresponding net-points $p(x)$ and $p(y)$ are distinct, which implies that the corresponding outgoing edges $e(x)$ and $e(y)$ are distinct as well. Hence $\omega(MST(X_A)) = \sum_{x \in A \setminus \{rt\}} \omega(e(x)) \geq \sum_{x \in A \setminus \{rt\}} \frac{rad(x)}{5}$. Finally, note that $rad(rt) \leq \omega(MST(X))$. It follows that $\sum_{x \in A} rad(x) = O(\omega(MST(X)))$.

Denote by $\tilde{\mathcal{H}}$ the spanner obtained from $\mathcal{H}$ by replacing the bipartite cliques with bipartite matchings, whenever possible. The above discussion shows that $\tilde{\mathcal{H}}$ achieves the desired lightness bound $\epsilon^{-O(d)}(k^2 + k \log n)$. Also, the degree of $\tilde{\mathcal{H}}$ is no greater than the degree of $\mathcal{H}$, and so it is in check too. Finally, we can guarantee that $\tilde{\mathcal{H}}$ will be a k-FT $(1+\epsilon)$-spanner using a simple adjustment, namely, connect the surrogate set $S(x)$ of each dirty leaf x via an *internal clique* (which includes an edge between any pair of points from $S(x)$). It is easy to see that this adjustment does not increase the degree and lightness by more than constants. Indeed, any point p may be a surrogate of at most $O(1)^{O(d)}$ leaves, and so its degree due to the internal cliques will increase by an additive factor of $O(1)^{O(d)} \cdot k$. In particular, this means that the number of edges due to the internal cliques is bounded above by $O(1)^{O(d)} \cdot kn$. By Observation 3.9, any such edge has weight at most $2(34 \cdot 5^0) = O(1)$, and so the total weight due to the internal cliques is at most $O(1)^{O(d)} \cdot kn \leq O(1)^{O(d)} \cdot k \cdot \omega(MST(X))$, where the last inequality follows from the fact that the minimum inter-point distance of X is equal to 1. Hence, the lightness will be in check as well.

Next, we show that for any pair $p, q \in X \setminus F$ of functioning points (where $|F| \leq k$), there is a $(1+\epsilon)$-spanner path in $\tilde{\mathcal{H}} \setminus F$. Consider the $(1+\epsilon)$-spanner path $\Pi_{p,q}$ between p and q in the basic spanner H that is guaranteed by Lemma 2.2, obtained by climbing up from the leaf nodes $(p, 0)$ and $(q, 0)$ to some j-level ancestors (p', j) and (q', j), respectively, which are connected by a cross edge $((p', j), (q', j))$.

Consider first the sub-path $\Pi_{p,p'}$ of $\Pi_{p,q}$ that climbs from $(p, 0)$ to (p', j). We argue that there is a short functioning path from p to some point in $S(p', j) \setminus F$. Note that p is a surrogate of every clean node along this path. The case when (p', j) is clean is immediate. Suppose that (p', j) is dirty, and consider the highest clean ancestor $x^{(0)}$ of $(p, 0)$ along $\Pi_{p,p'}$, and its dirty parent $x^{(1)} = \pi(x^{(0)})$. Observation 3.5 implies that p belongs to $S(x^{(0)})$. By Observation 3.4, $|S(x^{(1)}| = k+1$. There is a bipartite clique between $S(x^{(0)})$ and $S(x^{(1)})$ by the construction, and so p is connected to each of the $k+1$ points of $S(x^{(1)})$. We have disregarded the case where $x^{(1)} = (p, 0)$ is a dirty leaf. Recall that our adjustment added an internal clique for each dirty leaf. Hence, in this case p is connected to each of the other k points of $S(x^{(1)})$.

By Observation 3.4, all ancestors of $x^{(1)}$ are dirty. Let $\tilde{x}^{(1)}$ be the highest (dirty) ancestor of $x^{(1)}$ along $\Pi_{p,p'}$ with the same surrogate set as $x^{(1)}$, i.e., $S(\tilde{x}^{(1)}) = S(x^{(1)})$. (It is possible that $\tilde{x}^{(1)} = x^{(1)}$.)

Let $x^{(2)} = \pi(\tilde{x}^{(1)})$ be the parent of $\tilde{x}^{(1)}$, and let $\tilde{x}^{(2)}$ be the highest (dirty) ancestor of $x^{(2)}$ along $\Pi_{p,p'}$ with the same surrogate set as $x^{(2)}$, i.e., $S(\tilde{x}^{(2)}) = S(x^{(2)})$. (It is possible that $\tilde{x}^{(2)} = x^{(2)}$.) By definition, $S(\tilde{x}^{(2)}) = S(x^{(2)}) \neq S(\tilde{x}^{(1)}) = S(x^{(1)})$. Moreover, Claim 3.11 implies that $S(x^{(1)}) \cap S(x^{(2)}) = \emptyset$. By construction, $|S(x^{(1)})| = |S(x^{(2)})| = k+1$, and there is a perfect bipartite matching (of size $k+1$) between $S(\tilde{x}^{(1)}) = S(x^{(1)})$ and $S(x^{(2)}) = S(\tilde{x}^{(2)})$. Similarly, let $x^{(3)} = \pi(\tilde{x}^{(2)})$ be the parent of $\tilde{x}^{(2)}$, and let $\tilde{x}^{(3)}$ be the highest (dirty) ancestor of $x^{(3)}$ along $\Pi_{p,p'}$ with the same surrogate set as $x^{(3)}$, i.e., $S(\tilde{x}^{(3)}) = S(x^{(3)})$. By definition,

$$S(\tilde{x}^{(3)}) \;=\; S(x^{(3)}) \;\neq\; S(\tilde{x}^{(2)}) \;=\; S(x^{(2)}) \;\neq\; S(\tilde{x}^{(1)}) \;=\; S(x^{(1)}) \;\neq\; S(x^{(3)}).$$

Moreover, Claim 3.11 implies that

$$S(x^{(2)}) \cap S(x^{(3)}) \;=\; S(x^{(1)}) \cap S(x^{(3)}) \;=\; S(x^{(1)}) \cap S(x^{(2)}) \;=\; \emptyset.$$

By construction, $|S(x^{(2)})| = |S(x^{(3)})| = k+1$, and there is a perfect bipartite matching (of size $k+1$) between $S(\tilde{x}^{(2)}) = S(x^{(2)})$ and $S(x^{(3)}) = S(\tilde{x}^{(3)})$. It follows that there are $k+1$ vertex-disjoint paths between $S(x^{(1)})$ and $S(\tilde{x}^{(3)})$.

We continue this way until we reach $\tilde{x}^{(l)} = (p', j)$. By applying this argument inductively, we get $k+1$ vertex-disjoint paths between $S(x^{(1)})$ and $S(\tilde{x}^{(l)}) = S(p', j)$. Since $|F| \leq k$ (i.e., there are at most k faulty points), at least one of these paths must function. Recall also that p is connected to all points of $S(x^{(1)})$, which means that there is a path in $\tilde{\mathcal{H}} \setminus F$ between p and some functioning point $\tilde{p}$ of $S(p', j)$. Moreover, this functioning path is a $(1 + O(\epsilon))$-spanner path by Lemma 2.3, and we can reduce the stretch to $1 + \epsilon$ by scaling γ up by some constant. This proves the above assertion, but we are not done yet. In exactly the same way as above, it can be shown that there is a functioning $(1+\epsilon)$-spanner path between q and some functioning point $\tilde{q}$ of $S(q', j)$. However, the cross edge $(\tilde{p}, \tilde{q})$ need not be in our spanner $\tilde{\mathcal{H}}$. (This is because we use bipartite matchings rather than bipartite cliques, whenever possible.) Consequently, it may be impossible to glue these two functioning paths together via the edge $(\tilde{p}, \tilde{q})$. We can easily overcome this hurdle in the degenerate case where either (p', j) or (q', j) (or both) are clean. Indeed, suppose without loss of generality that (p', j) is clean. In this case p belongs to $S(p', j)$. Moreover, by construction, there is a bipartite clique between $S(p', j)$ and $S(q', j)$, and so p is connected via cross edges to all points of $S(q', j)$. If (q', j) is clean too, then there is a direct edge between p and q. Otherwise, we can use the above functioning $(1 + \epsilon)$-spanner path between q and $\tilde{q}$, and then get from $\tilde{q}$ to p via a direct edge. We thus obtain a short functioning path between p and q, as required. We will next show how to overcome the above hurdle in the more interesting case where both (p', j) and (q', j) are dirty.

The above argument handles the two sub-paths $\Pi_{p,p'}$ (between p and p') and $\Pi_{q,q'}$ (between q and q') of $\Pi_{p,q}$ separately. More specifically, it first builds a short functioning path for each of these two sub-paths of $\Pi_{p,q}$, and then it tries to glue these sub-paths together into a single functioning path using a possibly non-existing cross edge $(\tilde{p}, \tilde{q})$ (i.e., $(\tilde{p}, \tilde{q})$ may not belong to $\tilde{\mathcal{H}}$). Instead of breaking the path $\Pi_{p,q}$ into two sub-paths and handling them separately, we consider the entire path $\Pi_{p,q}$ and build for it a single functioning path in roughly the same way as we did for each of these two sub-paths.

As before, let $x^{(0)}$ be the highest clean ancestor of $(p, 0)$ along $\Pi_{p,p'}$, and let $x^{(1)} = \pi(x^{(0)})$ be its dirty parent. Similarly, let $y^{(0)}$ be the highest clean ancestor of $(q, 0)$ along $\Pi_{q,q'}$, and let $y^{(1)} = \pi(y^{(0)})$ be its dirty parent. Recall also that we defined $\tilde{x}^{(1)}$ as the highest (dirty) ancestor of $x^{(1)}$ along the sub-path $\Pi_{p,p'}$ with the same surrogate set as $x^{(1)}$. Now we define $\tilde{x}^{(1)}$ differently. More specifically, unlike the above argument, we would also like to consider nodes that belong to the other sub-path $\Pi_{q,q'}$ of $\Pi_{p,q}$. Let $\Pi_{x^{(1)},y^{(1)}} = (v_1 = x^{(1)}, v_2 = \pi(x^{(1)}), \ldots, v_t = (p', j), v_{t+1} = (q', j), \ldots, v_h = y^{(1)})$ be the sub-path of $\Pi_{p,q}$ from $x^{(1)}$ to $y^{(1)}$, where $1 \leq t < h \leq 2j+2$. Let i be the maximum index in $[h]$, such that $S(x^{(1)}) = S(v_i)$; we define $\tilde{x}^{(1)}$ as v_i. (Unlike before, now it is indeed possible that $\tilde{x}^{(1)}$ would belong to $\Pi_{q,q'}$.)

In the above argument, we defined $x^{(2)} = \pi(\tilde{x}^{(1)})$. This makes sense, as the objective was to get closer and closer to (p, j'). Here the objective is to get closer and closer to $y^{(1)}$. So we will define $x^{(2)} = \pi(\tilde{x}^{(1)})$ only in the case that $\tilde{x}^{(1)}$ belongs to $\Pi_{p,p'} \setminus (p', j)$. If $\tilde{x}^{(1)} = (p', j)$, then we define $x^{(2)} = (q', j)$. Finally, in the case that $\tilde{x}^{(1)}$ belongs to $\Pi_{q,q'}$, we define $x^{(2)}$ to be the child of $\tilde{x}^{(1)}$ along $\Pi_{q,q'}$.

In the same way we define $\tilde{x}^{(2)}, x^{(3)}$, and so forth. We continue this way until we reach $\tilde{x}^{(l)} = y^{(1)}$. In exactly the same way as before we get $k+1$ vertex-disjoint paths between $S(x^{(1)})$ and $S(\tilde{x}^{(l)}) = S(y^{(1)})$. Since $|F| \leq k$, at least one of these paths must function. Observe also that p and q are connected to all points of $S(x^{(1)})$ and $S(y^{(1)})$, respectively. (Recall that our adjustment added an internal clique for each dirty leaf. Hence, in the case where $x^{(1)} = (p, 0)$ (respectively, $y^{(1)} = (q, 0)$) is a dirty leaf, the point p (resp., q) is connected to each of the other k points of $S(x^{(1)})$ (resp., $S(y^{(1)})$).) Consequently, we obtain a path in $\tilde{\mathcal{H}} \setminus F$ between p and q. Moreover, this functioning path is a $(1+O(\epsilon))$-spanner path by Lemma 2.3, and we can reduce the stretch to $1+\epsilon$ by scaling γ up by some constant.

5 Diameter

In this section we show how to obtain diameter $O(\log n)$, without increasing any of the other parameters.

The basic idea was used before in [14, 44, 15], and involves shortcutting the *light subtrees* of the net-tree, i.e., those with distance scales less than $\frac{\Delta}{n}$, where $\Delta = \max_{u,v \in X} \delta(u, v)$ is the diameter of the metric X. This shortcutting is carried out using the 1-spanners of [45] (cf. Theorem 3 therein [45]).

Theorem 5.1 ([45]) *Let T be an arbitrary n-node tree. One can build in $O(n \log n)$ time, a 1-spanner for (the tree metric induced by) T with $O(n)$ edges, degree at most $\mathsf{deg}(T) + O(1)$ and diameter $O(\log n)$.*

Next, we would like to add shortcut edges to the spanner $\tilde{\mathcal{H}}$ of Section 4. The naïve approach would be to build the shortcut spanner of Theorem 5.1 for each of the light subtrees of the net-tree T, and then replace each shortcut edge by a bipartite matching if possible (i.e., if both endpoints of that edge are dirty), and by a bipartite clique otherwise. This would give rise to a k-FT $(1+\epsilon)$-spanner with logarithmic diameter. However, the degree and lightness would grow by too much, and would exceed the desired thresholds of $\epsilon^{-O(d)} \cdot k$ and $\epsilon^{-O(d)}(k^2 + k \log n)$, respectively.

In order to control the degree and lightness due to shortcut edges, we first *prune* some nodes from T, and only later proceed to shortcutting the light subtrees of the resulting *pruned tree* T^*. Then we proceed as before, replacing each shortcut edge by a bipartite matching (if possible) or a bipartite clique.

Our ultimate spanner $\mathcal{H}^*$ is obtained by adding the shortcut edges to the spanner $\tilde{\mathcal{H}}$ of Section 4.

We turn to describing how the pruned tree T^* is obtained.

- First, we remove all clean nodes from the net-tree. Indeed, shortcutting clean nodes is redundant, as the first dirty node on any $(1+\epsilon)$-spanner path is connected to its clean child on that path via a bipartite clique (i.e., there is a bipartite clique between the corresponding surrogate sets). Since the surrogate set of any clean node contains its entire descendant set (see Observation 3.5), one can go from any leaf point to all the surrogates of the relevant dirty node via a direct edge. In the degenerate case where the leaf itself is dirty, we have an internal clique over the surrogate set of that leaf (recall that the spanner $\tilde{\mathcal{H}}$ of Section 4 contains an internal clique over the surrogate set of each dirty leaf), and so one can still go from any leaf point to all the surrogates of that dirty leaf via a direct edge.

- Second, we go over the resulting tree (which contains only dirty nodes) top-down, looking for nodes x whose surrogate set is the same as that of its parent $\pi(x)$ and as that of every one of its dirty siblings. Such nodes are called *redundant*, and they are removed from the tree. When a node x and its siblings are removed from the tree, we connect $\pi(x)$ with its grandchildren. These grandchildren become the new children of $\pi(x)$, and they will be siblings in T^* (hereafter, *T^*-siblings*). However, they too may be redundant (if their surrogate set is the same as $\pi(x)$), and thus removed from the tree. This process may be repeated over and over, hence the children of $\pi(x)$ in the resulting pruned tree T^* may be of significantly smaller level than that of $\pi(x)$. That is, the parent x' of an i-level node x in the pruned tree T^* may be of level i', where $i \ll i'$. Notice that $S(x') \neq S(x)$. Moreover, for every j-level node z on the path between x and x' in the original tree T, where $j \in [i+1, i'-1]$, we have $S(z) = S(x')$. (Indeed, otherwise the node z would not be redundant;

since x is dirty, z will be dirty too by Observation 3.4, and so it would have to belong to T^*, which is a contradiction.) In other words, the edge (x, x') in the pruned tree T^* corresponds to a path $(x = v_i, v_{i+1}, v_{i+2}, \ldots, x' = v_{i'})$ in the original tree T, where all vertices $v_{i+1}, \ldots, v_{i'-1}$ are redundant and $S(v_{i+1}) = S(v_{i+2}) = \ldots = S(x') \neq S(x)$. In particular, this means that for any pair x, y of (dirty non-redundant) nodes in T^*, x is an ancestor of y in T^* if and only if x is an ancestor of y in the original net-tree T.

Note also that the degree of the resulting pruned tree T^* may be larger than the degree of the original tree T. However, by Fact 2.1, there can be only $O(1)^{O(d)}$ nodes with the same surrogate set at each level. This means that the degree of T^* will exceed the degree of T by at most a factor of $O(1)^{O(d)}$. Since the degree of the original tree T is $O(1)^{O(d)}$, it follows that the degree of the pruned tree T^* is at most $O(1)^{O(d)} \cdot O(1)^{O(d)} = O(1)^{O(d)}$, i.e., the degree does not grow by too much.

For a node x in T, denote by $\pi(x)$ (respectively, $\pi^*(x)$) its parent in the original tree T (resp., pruned tree T^*). If x does not belong to T^*, then $\pi^*(x)$ denotes the first ancestor of x in T which belongs to T^*.

The pruned tree T^* has some important advantages over the original net-tree T.

First, all nodes of T^* are dirty, and so any shortcut edge for T^* will translate into a bipartite matching rather than a bipartite clique. This property, however, is not required for achieving the desired bounds. That is, even if we are being wasteful and translate the shortcut edges into bipartite cliques, the desired bounds on the degree and lightness will still be in check; we shall address this issue in the sequel.

We use the fact that there are no clean and redundant nodes in T^* to prove the following lemma, which is critical for achieving the desired bounds on both the degree and the lightness.

Lemma 5.2 *Every point $p \in X$ may serve as a surrogate of at most $\epsilon^{-O(d)}$ nodes of T^*.*

Proof: Consider an arbitrary point $p \in X$, and let i be the first level in which p is a surrogate of some (dirty non-redundant) i-level node x in T^*. If x is the root of T^*, then we are done. We henceforth assume that x is not the root of T^*. Since x is non-redundant, its surrogate set must be different than either the surrogate set $S(\pi^*(x))$ of its parent $\pi^*(x)$ in T^* or the surrogate set $S(y)$ of some dirty T^*-sibling y of x. Define x' as x if it is a non-leech, or as the host of x otherwise. Note that x' is dirty and $S(x') = S(x)$. It is possible that x' is redundant, and thus does not belong to the pruned tree T^*.

We argue that $term(x')$ must be over until level $i + \tau + 3$. Note that the parent $\pi(x')$ of x' in T may be different than the parent $\pi^*(x')$ of x' in T^*; in this case $\pi(x')$ is a descendant of $\pi^*(x')$ in T. However, in either case $S(\pi^*(x')) = S(\pi(x'))$ holds by the construction. Similarly, we have $S(\pi^*(x)) = S(\pi(x))$.

In the case that $S(x') \neq S(\pi^*(x'))$, we have $S(x') \neq S(\pi(x'))$, which means that $term(x')$ is over at level i. We henceforth assume that $S(x') = S(\pi^*(x'))$, and so $S(x) = S(x') = S(\pi(x'))$.
Suppose first that $S(x) \neq S(\pi(x))$, which means that $S(\pi(x)) \neq S(\pi(x'))$, and so $\pi(x) \neq \pi(x'), x \neq x'$. In this case x is a leech of x'. By Observation 3.2, $\pi(x)$ and $\pi(x')$ are 24-friends, and so they are connected by a cross edge. Since x is dirty, $\pi(x)$ is also dirty by Observation 3.4, and we have $|S(\pi(x))| = k+1$. The fact that $S(\pi(x)) \neq S(\pi(x'))$ implies that the degree of the surrogates of $S(\pi(x'))$ due to non-redundant cross edges since their appointment reaches $k+1$ at level $i+1$ or before. It follows that the second phase of $term(x')$ will start at level $i+2$ or before. The second phase lasts precisely $\tau+2$ levels, and so $term(x')$ will be over until level $i + \tau + 3$.
Next, assume that $S(x) = S(\pi(x))$. By construction, $S(\pi(x)) = S(\pi^*(x))$, and so $S(x) = S(\pi^*(x))$. In this case it must hold that $S(x) \neq S(y)$, for some dirty T^*-sibling y of x. Since x and y are T^*-siblings, $S(\pi(x)) = S(\pi(y))$. (It is possible that $\pi(x) = \pi(y)$.) Observation 3.9 implies that $\pi(x)$ and $\pi(y)$ are 68-friends, and so x and y are 370-friends. Since x is a leech of x', it follows that x' and y are 394-friends, and so there is a cross edge between x' and y (assuming $\gamma \geq 394$). Since y is dirty, we have $|S(y)| = k+1$. Note also that $S(x') = S(x) \neq S(y)$. Hence the degree of the surrogates of $S(x')$ due to non-redundant cross edges since their appointment reaches $k+1$ at level i or before. Consequently, the second phase of $term(x')$ will start at level $i+1$ or before, and so $term(x')$ will be over until level $i + \tau + 2$.

We have shown that $term(x')$ must be over until level $i + \tau + 3$. Moreover, by Claim 3.11 and the construction, p cannot be a surrogate of any j-level node, for all $j \in [i + \tau + 4, \ell]$. In other words, p may only serve as a surrogate of j-level nodes, where $j \in [i, i+\tau+3]$. Since p may serve as a surrogate of at most $O(1)^{O(d)}$ nodes at each level of the original tree T, it will also serve as a surrogate of at most $O(1)^{O(d)}$ nodes at each level of the pruned tree T^*. We conclude that p may serve as a surrogate of at most $(\tau+4) \cdot O(1)^{O(d)} \leq \epsilon^{-O(d)}$ nodes of the pruned tree T^*, as required. ▮

Denote by $T_1^*, \ldots, T_m^*$ the light subtrees of the pruned tree T^*. Notice that the node sets of all the light subtrees are pairwise disjoint. Consider now an arbitrary light subtree T_i^* of T^*. Its degree $\mathsf{deg}(T_i^*)$ is at most $\mathsf{deg}(T^*) = O(1)^{O(d)}$. We use the 1-spanners of Theorem 5.1 to shortcut T_i^*, and then translate each non-redundant shortcut edge (x, y) into a perfect bipartite matching between the corresponding surrogate sets $S(x)$ and $S(y)$. Notice that such a matching between $S(x)$ and $S(y)$ increases the degree of each point in $S(x) \cup S(y)$ by only one unit. Any redundant shortcut edge (x, y) (with $S(x) = S(y)$) is simply disregarded. By Theorem 5.1, the shortcut edges increase the degree of each tree node of T_i^* by at most $\mathsf{deg}(T_i^*) + O(1) = O(1)^{O(d)}$ units. Also, Lemma 5.2 shows that every point may serve as a surrogate of at most $\epsilon^{-O(d)}$ nodes of T^*. It follows that the degree of any point (due to the bipartite matchings that correspond to the non-redundant shortcut edges) is at most $\epsilon^{-O(d)}$, and so the degree of the spanner $\mathcal{H}^*$ will be in check. In fact, even if we are being wasteful and translate the non-redundant shortcut edges into bipartite cliques, the degree of $\mathcal{H}^*$ will still be bounded above by the desired threshold of $\epsilon^{-O(d)} \cdot k$.

By Lemma 5.2, the number of nodes in the pruned tree T^* is bounded above by $\epsilon^{-O(d)} \cdot n$. As mentioned, the node sets of all the light subtrees of T^* are pairwise disjoint. Hence, by Theorem 5.1, the total number of all shortcut edges is bounded above by $O(\epsilon^{-O(d)} \cdot n) = \epsilon^{-O(d)} \cdot n$. Since the distance scale of each light subtree is at most $\frac{\Delta}{n}$ (and as the distance scales decrease geometrically with the level), the weight of any shortcut edge is at most $O(\frac{\Delta}{n})$. Moreover, each of the $k+1$ edges of the bipartite matching that replaces a shortcut edge is also bounded by $O(\frac{\Delta}{n})$. Note also that $\Delta \leq \omega(MST(X))$. It follows that the total weight of the bipartite matchings that correspond to all the non-redundant shortcut edges is at most $\epsilon^{-O(d)} \cdot n \cdot O(\frac{\Delta}{n})(k+1) \leq \epsilon^{-O(d)} \cdot k \cdot \omega(MST(x))$, and so the lightness of the spanner $\mathcal{H}^*$ will be in check. In fact, even if we are being wasteful and translate the non-redundant shortcut edges into bipartite cliques, the lightness of $\mathcal{H}^*$ will still be bounded above by the desired threshold of $\epsilon^{-O(d)}(k^2 + k \log n)$.

Finally, we show that $\mathcal{H}^*$ is a k-FT $(1+\epsilon)$-spanner with diameter $O(\log n)$. Consider an arbitrary pair $p, q \in X \setminus F$ of functioning points (where $|F| \leq k$), and let $\Pi_{p,q}$ be the $(1+\epsilon)$-spanner path between p and q in the spanner H that is guaranteed by Lemma 2.2. Recall that $\Pi_{p,q}$ is obtained as the union of the two sub-paths $\Pi(p, p')$ and $\Pi(q, q')$ between $(p, 0)$ and (p', j) and between $(q, 0)$ and (q', j), respectively, which are glued together via the j-level cross edge between (p', j) and (q', j). Note that some of the nodes along $\Pi_{p,q}$ may not belong to T^*. Let $\Pi^*_{p,p'}$ be the path obtained from $\Pi_{p,p'}$ by (i) replacing each node x along $\Pi_{p,p'}$ which does not belong to T^* with $\pi^*(x)$ (i.e., with the first ancestor of x in T that belongs to T^*), and (ii) removing duplicates, i.e., leaving only the last occurrence of each node in that path. Let x^* be the last node in $\Pi^*_{p,p'}$, and observe that $\Pi^*_{p,p'}$ is almost a sub-path of $\Pi_{p,p'}$. More specifically, if (p', j) belongs to T^*, then $\Pi^*_{p,p'}$ is a sub-path of $\Pi_{p,p'}$, with $x^* = (p', j)$. Otherwise the last node x^* along $\Pi^*_{p,p'}$ is $\pi^*(p', j)$, and the path $\Pi^*_{p,p'} \setminus \pi^*(p', j)$ is a sub-path of $\Pi_{p,p'}$. We define $\Pi^*_{q,q'}$ in the same way, and denote the last node in $\Pi^*_{q,q'}$ by y^*. Let $\Pi^*_{p,q} = \Pi^*_{p,p'} \circ (x^*, y^*) \circ \Pi^*_{q,q'}$ be the union of the two paths $\Pi^*_{p,p'}$ and $\Pi^*_{q,q'}$, which are glued together via the edge (x^*, y^*). Observe that x^* and y^* may be different than (p', j) and (q', j), respectively; in particular, the edge (x^*, y^*) may not belong to the spanner H. Nevertheless, we must have $S(x^*) = S(p', j)$ and $S(y^*) = S(q', j)$ by the construction. Consequently, the possible differences between x^* and (p', j) and between y^* and (q', j) are insignificant, as x^* and y^* can play the role of (p', j) and (q', j), respectively. That is, the bipartite clique or matching that should replace the possibly non-existing edge (x^*, y^*) is incarnated in the bipartite clique or matching that replaces the j-level cross edge between (p', j) and (q', j).

Observe that the path $\Pi^*_{p,q}$ may contain many edges. This is where the shortcut edges come in handy. That is, we can use the shortcut edges to reduce the number of edges in the path $\Pi^*_{p,q}$ to $O(\log n)$, without increasing the weight of the path. The proof for this assertion is simple, and follows similar lines as those in the works of [14, 44, 15]. (The main idea behind this proof is that there are only $O(\log n)$ levels with distance scales at least $\frac{\Delta}{n}$. In other words, by removing the light subtrees $T_1^*, \ldots, T_m^*$ from the pruned tree T^*, we obtain a tree $\bar{T}$ with only $O(\log n)$ levels. To maneuver over the (possibly many) remaining levels of $\bar{T}$ we use the shortcut edges.) We will henceforth assume that $\Pi^*_{p,q}$ contains only $O(\log n)$ edges.

In Section 4 we used a sub-path $\Pi_{x^{(1)},y^{(1)}}$ of $\Pi_{p,q}$ to show that there are $k+1$ vertex-disjoint paths between $S(x^{(1)})$ and $S(y^{(1)})$, where $x^{(1)}$ (respectively, $y^{(1)}$) stands for the first dirty node along $\Pi(p,p')$ (resp., $\Pi(q,q')$). By using the path $\Pi^*_{p,q}$ instead of $\Pi_{x^{(1)},y^{(1)}}$, we can show in exactly the same way that there are $k+1$ vertex-disjoint paths between the surrogate sets $S(x_1)$ and $S(y_1)$ of the first and last nodes x_1 and y_1 along $\Pi^*_{p,q}$, respectively. Moreover, since $\Pi^*_{p,q}$ contains only $O(\log n)$ edges, all these paths have at most $O(\log n)$ edges. The fact that $|F| \leq k$ implies that at least one of these paths must function. Finally, observe that p and q are connected to all points of $S(x_1)$ and $S(y_1)$, respectively. It follows that there is a $(1+\epsilon)$-spanner path in $\mathcal{H}^* \setminus F$ between p and q that consists of only $O(\log n)$ edges.

6 Running Time

Our construction uses the net-tree spanner of [40, 30] and the 1-spanners of [45] as black-boxes. The construction of [40, 30] (respectively, [45]) can be implemented within time $\epsilon^{-O(d)} \cdot n \log n$ (resp., $O(n \log n)$).

A central ingredient in the net-tree spanner of [40, 30] is the underlying net-tree, which is based on a sophisticated hierarchical partition from [11]. To build the net-tree and the underlying spanner efficiently, it is important to be able to concentrate on only $\epsilon^{-O(d)} \cdot n$ *useful* tree nodes. In particular, a node is called *lonely* if it has exactly one child in T (which corresponds to the same net-point as the parent); otherwise it is *non-lonely*. Following [40, 30, 15] and other works in this context, a long chain of lonely nodes will be represented implicitly for efficiency reasons; implementation details can be found in [30].

Our algorithm from Section 3.1 (see in particular Section 3.1.4) traverses the net-tree bottom-up. The sets $S(\cdot), D(\cdot), F(\cdot), R(\cdot)$ are computed for the i-level (useful) nodes, only after we have computed these sets for the nodes at lower levels $j \in [0, i-1]$. The computation of these sets is carried out via Procedure $\mathit{ComputeSets}_{(i)}$, which makes heavy use of the i-level cross edges that we are given from the net-tree spanner of [40, 30]. Notice that each tree node x is incident on only $\epsilon^{-O(d)}$ cross edges and only $O(1)^{O(d)}$ tree edges. To compute the sets $S(x), D(x), F(x), R(x)$ for an i-level node x, we need to gather information from x's neighbors y either due to cross edges or due to tree edges (more specifically, children), where most of the information about such neighbors y is held in the corresponding sets $S(y), D(y), F(y), R(y)$ of y. With this information at hand, it is rather simple to compute the sets $S(x), D(x), F(x), R(x)$. In particular, since a potential host for a node is a 24-friend of that node, there is a cross edge between a node and all its potential hosts; hence finding a potential host can be carried out efficiently.

The descendant sets $D(\cdot)$ of all tree nodes can be computed in a straightforward way within $\epsilon^{-O(d)} \cdot n$ time. It is more difficult to compute the other sets $S(\cdot), F(\cdot), R(\cdot)$ efficiently. In particular, note that Procedure $\mathit{ComputeSets}_{(i)}$ upper bounds the sizes of the surrogate sets $S(\cdot)$ and the friend sets $F(\cdot)$ by $2k+1$ and $3k+3$, respectively. Upper bounding the size of $S(\cdot)$ and $F(\cdot)$ by $\epsilon^{-O(d)} \cdot k$ is critical in order to perform efficiently. Similarly, we also need to bound the size of the reserve sets $R(\cdot)$ by $\epsilon^{-O(d)} \cdot k$. Consider an arbitrary node x. We argue that it suffices to keep track of the clean points that were added to the reserve sets of either x or its descendants in the last τ levels. Indeed, a point p that belongs to the reserve set $R(y)$ of some i-level node y will be a 10-friend of the $(i+\tau)$-level ancestor y' of y by Claim 3.3. Assuming p belongs to $F(y')$, there is no reason to store it in $R(y')$ as well. If some node $\tilde{y}$ on the path in T between y and y' satisfies $|F(\tilde{y})| = 3k+3$, then p may not belong to $F(y')$. (This is the only case in which p may not belong to $F(y')$.) However, in this case we should have enough points in $F(y')$. To be on the safe side, besides the points that were added to the reserve sets of either x or its descendants in the last τ levels, we may store in $R(x)$ additional $O(k)$ points that were added to the reserve sets of x's descendants before the last τ levels; as mentioned, such points are 10-friends of x. In this way we always keep track of enough clean 10-friends of a node. By construction, in each level at most $\epsilon^{-O(d)} \cdot k$ new points are added to the reserve set of any node. Since we only keep track of the last τ levels (and at most $O(k)$ additional points), it follows that the size of the reserve sets $R(\cdot)$ can be upper bounded by $\tau \cdot \epsilon^{-O(d)} \cdot k + O(k) \leq \epsilon^{-O(d)} \cdot k$. Given that the size of all sets $S(\cdot), F(\cdot), R(\cdot)$ is bounded above by

$\epsilon^{-O(d)} \cdot k$, it is easy to see that the overall time needed to compute the sets $S(x), F(x), R(x)$ for all useful nodes x in the tree will be $(\epsilon^{-O(d)} \cdot k) \cdot (\epsilon^{-O(d)} \cdot n) \le \epsilon^{-O(d)} \cdot kn$. Having computed the surrogate sets of all nodes, we replace each edge of the net-tree spanner H of [40, 30] by a bipartite clique between the corresponding surrogate sets. The time needed to replace all edges of the spanner H by bipartite cliques is asymptotically the same as the number of edges in the resulting FT spanner $\mathcal{H}$, which is $\epsilon^{-O(d)} \cdot kn$. It follows that the overall time required to build the spanner $\mathcal{H}$ from Section 3 is $\epsilon^{-O(d)} \cdot (n \log n + kn)$.

The transformation described in Section 4, which translates each bipartite clique into a bipartite matching whenever possible (i.e., when both endpoints of the corresponding edge in the spanner H are dirty), can be easily implemented within time $\epsilon^{-O(d)} \cdot kn$. Building the internal cliques for all dirty leaves requires another amount of $\epsilon^{-O(d)} \cdot kn$ time. Consequently, the spanner $\tilde{\mathcal{H}}$ from Section 4 can be implemented within time $\epsilon^{-O(d)} \cdot (n \log n + kn)$.

Finally, the construction of the pruned tree T^* in Section 5 can be carried out within time $\epsilon^{-O(d)} \cdot n$. By Theorem 5.1, shortcutting the light subtrees of T^* requires $\epsilon^{-O(d)} \cdot n \log n$ time. We conclude that the overall running time required to build the ultimate spanner $\mathcal{H}^*$ from Section 5 (which proves Theorem 1.1) is at most $\epsilon^{-O(d)} \cdot (n \log n + kn)$.

To summarize: The spanner $\mathcal{H}^*$ of Section 5 proves Theorem 1.1.

Acknowledgments

The author is grateful to Michael Elkin for helpful and timely discussions.

References

[1] I. Abraham, Y. Bartal, and O. Neiman. Advances in metric embedding theory. *Advances in Mathematics*, 228(6):30263126, 2011.

[2] I. Abraham and D. Malkhi. Compact routing on Euclidian metrics. In *Proc. of 23rd Annual Symp. on Principles of Distributed Computing*, pages 141–149, 2004.

[3] P. K. Agarwal, Y. Wang, and P. Yin. Lower bound for sparse Euclidean spanners. In *Proc. of 16th SODA*, pages 670–671, 2005.

[4] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.

[5] S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. H. M. Smid. Euclidean spanners: short, thin, and lanky. In *Proc. of 27th STOC*, pages 489–498, 1995.

[6] S. Arya, D. M. Mount, and M. H. M. Smid. Randomized and deterministic algorithms for geometric spanners of small diameter. In *Proc. of 35th FOCS*, pages 703–712, 1994.

[7] S. Arya and M. H. M. Smid. Efficient construction of a bounded degree spanner with low weight. In *Proc. of 2nd ESA*, pages 48–59, 1994.

[8] P. Assouad. Plongements lipschitziens dans $\mathbb{R}^n$. *Bull. Soc. Math. France*, 111(4):429448, 1983.

[9] Y. Bartal, L. Gottlieb, and R. Krauthgamer. The traveling salesman problem: low-dimensionality implies a polynomial time approximation scheme. In *Proc. of 44th STOC*, pages 663–672, 2012.

[10] P. B. Callahan and S. R. Kosaraju. A decomposition of multi-dimensional point-sets with applications to k-nearest-neighbors and n-body potential fields. In *Proc. of 24th STOC*, pages 546–556, 1992.

[11] H. T.-H. Chan and A. Gupta. Small hop-diameter sparse spanners for doubling metrics. In *Proc. of 17th SODA*, pages 70–78, 2006.

[12] H. T.-H. Chan, A. Gupta, B. M. Maggs, and S. Zhou. On hierarchical routing in doubling metrics. In *Proc. of 16th SODA*, pages 762–771, 2005.

[13] T.-H. H. Chan, M. Li, and L. Ning. Sparse fault-tolerant spanners for doubling metrics with bounded hop-diameter or degree. In *ICALP (1)*, pages 182–193, 2012.

[14] T.-H. H. Chan, M. Li, and L. Ning. Incubators vs zombies: Fault-tolerant, short, thin and lanky spanners for doubling metrics. *Technical Report, CoRR abs/1207.0892*, July, 2012.

[15] T.-H. H. Chan, M. Li, L. Ning, and S. Solomon. New doubling spanners: Better and simpler. In *Proc. of 40th ICALP*, 2013 (to appear).

[16] B. Chandra, G. Das, G. Narasimhan, and J. Soares. New sparseness results on graph spanners. In *Proc. of 8th SOCG*, pages 192–201, 1992.

[17] D. Z. Chen, G. Das, and M. H. M. Smid. Lower bounds for computing geometric spanners and approximate shortest paths. *Discrete Applied Mathematics*, 110(2-3):151–167, 2001.

[18] L. P. Chew. There is a planar graph almost as good as the complete graph. In *Proc. of 2nd SOCG*, pages 169–177, 1986.

[19] K. L. Clarkson. Approximation algorithms for shortest path motion planning. In *Proc. of 19th STOC*, pages 56–65, 1987.

[20] K. L. Clarkson. Nearest neighbor queries in metric spaces. *Discrete Comput. Geom.*, 110(1):6393, 1999.

[21] A. Czumaj and H. Zhao. Fault-tolerant geometric spanners. In *Proc. of 19th SoCG*, pages 1–10, 2003.

[22] G. Das, P. J. Heffernan, and G. Narasimhan. Optimally sparse spanners in 3-dimensional Euclidean space. In *Proc. of 9th SOCG*, pages 53–62, 1993.

[23] G. Das and G. Narasimhan. A fast algorithm for constructing sparse Euclidean spanners. In *Proc. of 10th SOCG*, pages 132–139, 1994.

[24] Y. Dinitz, M. Elkin, and S. Solomon. Low-light trees, and tight lower bounds for Euclidean spanners. *Discrete & Computational Geometry*, 43(4):736–783, 2010.

[25] M. Elkin and S. Solomon. Optimal Euclidean spanners: really short, thin and lanky. In *STOC*, 2013 (to appear).

[26] J. Gao, L. J. Guibas, and A. Nguyen. Deformable spanners and applications. In *Proc. of 20th SoCG*, pages 190–199, 2004.

[27] L. Gottlieb. Private communication, March 2013.

[28] L. Gottlieb, A. Kontorovich, and R. Krauthgamer. Efficient regression in metric spaces via approximate lipschitz extension. *CoRR*, abs/1111.4470, 2011.

[29] L. Gottlieb and L. Roditty. Improved algorithms for fully dynamic geometric spanners and geometric routing. In *Proc. of 19th SODA*, pages 591–600, 2008.

[30] L. Gottlieb and L. Roditty. An optimal dynamic spanner for doubling metric spaces. In *Proc. of 16th ESA*, pages 478–489, 2008. Another version of this paper is available via `http://cs.nyu.edu/~adi/spanner2.pdf`.

[31] J. Gudmundsson, C. Levcopoulos, and G. Narasimhan. Fast greedy algorithms for constructing sparse geometric spanners. *SIAM J. Comput.*, 31(5):1479–1500, 2002.

[32] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *Proc. of 44th FOCS*, page 534543, 2003.

[33] S. Har-Peled and M. Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM J. Comput.*, 35(5):1148–1184, 2006.

[34] S. Kapoor and X. Li. Efficient construction of spanners in d-dimensions. *Technical Report, CoRR abs/1303.7217*, March, 2013. Corresponds to an unpublished manuscript (the first draft is from November 2007), which is currently under review.

[35] J. M. Keil and C. A. Gutwin. Classes of graphs which approximate the complete Euclidean graph. *Discrete & Computational Geometry*, 7:13–28, 1992.

[36] R. Krauthgamer and J. R. Lee. Navigating nets: Simple algorithms for proximity search. In *Proc. of 15th SODA*, page 791801, 2004.

[37] C. Levcopoulos, G. Narasimhan, and M. H. M. Smid. Efficient algorithms for constructing fault-tolerant geometric spanners. In *Proc. of 30th STOC*, pages 186–195, 1998.

[38] T. Lukovszki. New results of fault tolerant geometric spanners. In *Proc. of 6th WADS*, pages 193–204, 1999.

[39] G. Narasimhan and M. Smid. *Geometric Spanner Networks.* Cambridge University Press, 2007.

[40] L. Roditty. Fully dynamic geometric spanners. In *Proc. of 23rd SoCG*, pages 373–380, 2007.

[41] J. S. Salowe. Construction of multidimensional spanner graphs, with applications to minimum spanning trees. In *Proc. of 7th SoCG*, pages 256–261, 1991.

[42] M. H. M. Smid. The weak gap property in metric spaces of bounded doubling dimension. In *Proc. of Efficient Algorithms*, pages 275–289, 2009.

[43] S. Solomon. From hierarchical partitions to hierarchical covers: Optimal fault-tolerant spanners for doubling metrics. Available via `http://cs.bgu.ac.il/~shayso/optFTspanner.pdf`.

[44] S. Solomon. Fault-tolerant spanners for doubling metrics: Better and simpler. *Technical Report, CoRR abs/1207.7040*, July, 2012.

[45] S. Solomon and M. Elkin. Balancing degree, diameter and weight in Euclidean spanners. In *Proc. of 18th ESA*, pages 48–59, 2010.

[46] K. Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In *Proc. of 36th STOC*, page 281290, 2004.

[47] P. M. Vaidya. A sparse graph almost as good as the complete graph on points in k dimensions. *Discrete & Computational Geometry*, 6:369–381, 1991.